Structuration de bases multimédia pour une exploration visuelle

Nicolas Voiron

UNIVERSITÉ GRENOBLE ALPES

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

Spécialité : **STIC Traitement de l'Information**

Arrêté ministériel : 7 août 2006

Présentée par

Nicolas VOIRON

Thèse dirigée par **Patrick LAMBERT**

et co-encadrée par **Alexandre BENOIT**

préparée au sein **du laboratoire LISTIC (Université Savoie Mont Blanc)** et de **l'École Doctorale SISEO**

Structuration de bases multimédia pour une exploration visuelle.

Thèse soutenue publiquement le **18 décembre 2015**,
devant le jury composé de :

M. Liming CHEN
Professeur, École Centrale de Lyon, Président
M. Philippe GOSSELIN
Professeur, ENSEA, Rapporteur
M. Nicolas LABROCHE
Maître de Conférences HDR, Université François Rabelais de Tours, Rapporteur
M. Bogdan IONESCU
Associate Professor Habilité, LAPI Bucarest, Examinateur
M. Philippe JOLY
Professeur, Université Toulouse III - Paul Sabatier, Examinateur
M. Patrick LAMBERT
Professeur, Université Savoie Mont Blanc, Directeur de thèse
M. Alexandre BENOIT
Maître de Conférences, Université Savoie Mont Blanc, Co-Encadrant de thèse

Résumé : La forte augmentation du volume de données multimédia impose la mise au point de solutions adaptées pour une exploration visuelle efficace des bases multimédia. Après avoir examiné les processus de visualisation mis en jeu, nous remarquons que ceci demande une structuration des données. L'objectif principal de cette thèse est de proposer et d'étudier ces méthodes de structuration des bases multimédia en vue de leur exploration visuelle.

Nous commençons par un état de l'art détaillant les données et les mesures que nous pouvons produire en fonction de la nature des variables décrivant les données. Suit un examen des techniques de structuration par projection et classification. Nous présentons aussi en détail la technique du Clustering Spectral sur laquelle nous nous focaliserons ensuite.

Notre première réalisation est une méthode originale de production et fusion de métriques par corrélation de rang. Nous testons cette première méthode sur une base multimédia issue de la vidéothèque d'un festival de films. Nous continuons ensuite par la mise au point d'une méthode de classification supervisée par corrélation que nous testons avec les données vidéos d'un challenge de la communauté multimédia. Ensuite nous nous focalisons sur les techniques du Clustering Spectral. Nous testons une technique de Clustering Spectral supervisée que nous comparons aux techniques de l'état de l'art. Et pour finir nous examinons des techniques du Clustering Spectral semi-supervisé actif. Dans ce contexte, nous proposons et validons des techniques de propagation d'annotations et des stratégies permettant d'améliorer la convergence de ces méthodes de classement.

Abstract : The large increase in multimedia data volume requires the development of effective solutions for visual exploration of multimedia databases. After reviewing the visualization process involved, we emphasis the need of data structuration. The main objective of this thesis is to propose and study clustering and classification of multimedia database for their visual exploration.

We begin with a state of the art detailing the data and the metrics we can produce according to the nature of the variables describing each document. Follows a review of the projection and classification techniques. We also present in detail the Spectral Clustering method.

Our first contribution is an original method that produces fusion of metrics using rank correlations. We validate this method on an animation movie database coming from an international festival. Then we propose a supervised classification method based on rank correlation. This contribution is evaluated on a multimedia challenge dataset. Then we focus on Spectral Clustering methods. We test a supervised Spectral Clustering technique and compare to state of the art methods. Finally we examine active semi-supervised Spectral Clustering methods. In this context, we propose and validate constraint propagation techniques and strategies to improve the convergence of these active methods.

Mots clefs : Structuration, bases multimédia, production de métrique, corrélation de rang, classification, Clustering Spectral, supervision, semi-supervision.

Remerciements

Cette thèse a été menée au sein du Laboratoire d'Informatique, Systèmes et Traitement de l'Information et de la Connaissance (LISTIC) de l'Université Savoie Mont Blanc. L'Université m'a accordé un aménagement de service pour me permettre de réaliser cette thèse. Mes premiers remerciements vont donc naturellement à cette institution.

Je remercie monsieur Philippe Gosselin et monsieur Nicolas Labroche d'avoir bien voulu rapporter mes travaux ainsi que les membres du jury monsieur Liming Chen, monsieur Bogdan Ionescu et monsieur Philippe Joly pour leurs remarques et suggestions concernant ce manuscrit et plus généralement sur mes travaux de thèse.

La soutenance d'une thèse est un événement qui clôture une étape d'une expérience qui est très enrichissante. Ainsi je tiens à remercier chaleureusement messieurs Patrick Lambert et Alexandre Benoit qui m'ont accompagné tout au long de cette thèse

Je souhaite remercier l'ensemble du laboratoire de m'avoir accueilli au sein du LISTIC.

Un merci tout particulier à tous les collègues de l'IUT d'Annecy qui m'ont soutenu pendant mes travaux de thèse.

Mes derniers remerciements vont tout naturellement à ma famille qui m'a soutenu et qui m'a permis d'aller jusqu'au bout de ce projet de thèse.

Table des matières

Liste des tableaux

Table des figures

Chapitre 1

Introduction

Dans beaucoup de domaines, la quantité de données multimédia augmentant fortement, il est nécessaire de disposer d'outils de recherche et de navigation adaptés pour pouvoir effectuer des explorations visuelles de ces bases multimédia qui soient efficaces.

Sur internet, les moteurs de recherche permettent de parcourir les données multimédia disponibles à l'aide d'une requête textuelle complétée parfois par une recherche par média semblables. Les résultats de recherche sont souvent ordonnés par pertinence décroissante. Cette pertinence peut prendre en compte ou non des informations personnelles que nous communiquons à l'aide de profils. L'exploration en devient personnalisée mais ceci nécessite des mécanismes et des techniques élaborées encore en cours de développement.

Il existent de nombreuses bases multimédia autres que celles rendues publiques sur internet. Des organisations comme les festivals de films à thèmes se constituent des médiathèques conséquentes. Dans ce contexte, les explorations visuelles peuvent être artistiques et correspondre à des balades ludiques dans la base multimédia. Cependant la plupart des autres explorations visuelles correspondent plutôt à un besoin de parcours intelligible et de recherche de média spécifique. Plus éloignées des loisirs créatifs, de nombreuses entreprises et institutions gèrent des archives multimédia volumineuses. Par exemple, dans le domaine médical, chaque examen de radiologie génère plusieurs centaines d'images numérisées. Pour un seul centre de radiologie, la masse annuelle de données multimédia produite est actuellement de plusieurs téraoctets. La mise à disposition de ces données aux radiologues, ne serait-ce que pour échanger des avis, oblige les centres à mettre à leur disposition des solutions d'exploration visuelle adaptées.

Outre les problématiques de stockage et d'accès aux données, le point qui est particulièrement important pour l'exploration visuelle de données multimédia est la visualisation qui doit être adaptée aux données et aux utilisateurs. Dans cette thèse, nous avons pris comme point d'entrée la finalité première des applications logicielles : la réponse aux besoins de l'utilisateur. Dans un contexte d'exploration de données multimédia, nous définissons en premier lieu la visualisation en fonction des besoins de l'utilisateur, sans tenir compte des éventuelles contraintes et structures intrinsèques des données explorées.

La visualisation étant définie indépendamment des données, il nous reste à mettre au point un processus qui à partir des données brutes permet d'aboutir à la visualisation. C'est

ce processus de visualisation que nous commençons par détailler. Fournir une liste de média non ordonnée et sans structure serait totalement inadapté face au volume de données considérées. Nous pressentons tout de suite que pour passer de données brutes à une visualisation permettant une exploration visuelle efficace, il est nécessaire de structurer les données de manière adéquate. Or, pour structurer les données multimédia, il faut commencer par les caractériser. Ceci consiste principalement en une extraction de descripteurs pertinents et compacts. Beaucoup de travaux existent sur les images statiques. Pour les vidéos, la quantité de données est beaucoup plus grande et l'analyse est beaucoup plus complexe. Dans tous les cas, ces descripteurs sont de plus bas niveau que ce qui est attendu par les utilisateurs. Cependant, on peut espérer en obtenir depuis plusieurs sources : les données brutes et les métadonnées que l'on peut espérer combiner pour résoudre le problème.

Les descripteurs étant extraits, nous avons une première structure souvent inexploitable. Seuls des descripteurs intelligibles et en faible nombre peuvent être directement exploités. Par exemple, pour une base de films, si l'on a quelques descripteurs de type métadonnée (l'année, le producteur, le réalisateur, le pays...), il est possible de proposer des filtres selon une année ou un réalisateur et obtenir une application adaptée au besoin d'exploration de l'utilisateur. Cependant, ceci ne permettra pas de répondre à des questions plus fines telles que « retrouver la séquence d'un film dans laquelle... ». C'est dans ce type de besoins que les descripteurs bas niveau peuvent répondre au problème, cependant ils ne sont pas exploitables directement par un utilisateur. Il convient donc de restructurer la base. C'est à dire d'obtenir d'autres structures à l'aide de méthode dite de structuration. Ces méthodes peuvent être des projections ou des classifications. Les projections permettent de diminuer la dimension des données et peuvent par exemple produire des placements 2D qui permettent d'avoir un affichage planaire des données. Les classifications permettent de regrouper les données qui se ressemblent et d'avoir un affichage des données par paquets. Ces méthodes peuvent intégrer de la connaissance pour obtenir de meilleurs performances ou générer des structures personnalisées en fonction par exemple d'un profil utilisateur.

Ce sont ces techniques de structuration appliquées aux bases multimédia qui sont le cœur de cette thèse nommée « Structuration de bases multimédia pour une exploration visuelle ».

La suite de ce document s'organise de la façon suivante : au chapitre 2, nous examinons le processus de visualisation pour mettre en évidence les besoins de structuration. Nous présentons les modèles de visualisation qui font référence et leurs différentes étapes. Nous résumons les visualisations existantes qui sont détaillées dans l'annexe A. Nous finissons par synthétiser tous ces aspects dans notre modèle de visualisation et nous montrons comment il structure nos réalisations détaillées dans les derniers chapitres.

Au chapitre 3, dans l'état de l'art, nous voyons comment travailler les données et produire des métriques. Puis nous examinons quelles sont les principales méthodes de projection et nous présentons des techniques de classification automatiques, supervisées et semi-supervisées. Nous présentons aussi comment évaluer le résultat des classifications grâce aux critères de qualité et de comparaison des partitions. Et pour finir, nous détaillons la méthode du Clustering Spectral et la théorie spectrale des graphes qu'elle utilise.

Au chapitre 4, nous développons dans les premiers paragraphes comment à partir d'une

base de données multimédia pour laquelle nous disposons d'une vérité terrain par paires, il est possible de sélectionner les descripteurs les plus pertinents. Nous présentons ensuite une méthode originale de fusion des descripteurs basée sur les coefficients de corrélation de rang. En fin de chapitre, nous développons et évaluons une méthode de classification par corrélation.

Au chapitre 5, nous continuons à investiguer sur les techniques de classification en nous intéressant aux techniques de Clustering Spectral. Pour améliorer les résultats, nous examinons comment ajouter de la supervision à cette méthode. Puis nous voyons comment les techniques de Clustering Spectral peuvent devenir supervisées et nous mettons au point des procédés itératifs pour obtenir un Clustering Spectral semi-supervisé efficace.

Chapitre 2

Le processus de visualisation et les besoins de structuration

Résumé : Dans ce chapitre nous nous intéressons à l'exploration visuelle d'une base multimédia. Nous examinons le contexte de la visualisation et nous montrons que les visualisations existantes sont nombreuses, variées et bien décrites. Nous présentons un processus d'élaboration des visualisations qui commence avec les données brutes. Pour mieux guider sa mise en œuvre, nous détaillons les différentes étapes de ce processus qui commence par une structuration des données pour ensuite sélectionner une représentation des données afin de finalement pouvoir les visualiser. Le principal apport de ce chapitre est de mettre en évidence les différents besoins en techniques de structuration des données. Ce sont ces besoins de structuration que nous développons dans tous les autres chapitres de cette thèse.

Proposer une exploration visuelle d'une base multimédia implique tout d'abord de proposer une visualisation adaptée aux données et surtout aux utilisateurs finaux. Il convient donc de commencer par préciser ce qu'est une visualisation et comment en produire une. Explorer visuellement une base documentaire nous amène directement dans le champ de la visualisation d'information souvent nommée « InfoVis » ou « IV ».

La visualisation est l'« action de rendre visible un phénomène qui ne l'est pas ». Pour la communauté de la visualisation d'information, la problématique de la visualisation peut se résumer à cette question : « comment représenter un grand nombre d'informations sur un écran ? ». En 2006, au cours d'une table ronde intitulée « Is there science in visualization ? » [Jankun-Kelly *et al.*, 2006], Wes Bethel explique que beaucoup dans les « sciences dures » voient l'informatique comme une science parvenue et manquant de rigueur scientifique. Il précise que pour le champ de la visualisation, c'est encore plus significatif et qu'il y a un « fossé de crédibilité » entre la communauté de la visualisation d'information et une partie de sa « clientèle » scientifique [Villerd, 2008].

Dans un processus de visualisation, les données de la base documentaire ont besoin d'être adaptées ou modifiées afin d'obtenir un structure qui permette de facilement représenter puis visualiser les données. Pour obtenir un carte bi ou tridimensionnelle des données, il convient de projeter les données de leur espace souvent de grande dimension, vers un espace 2D ou 3D. Il existe d'autres structures intéressantes comme les structures orientées liaisons, dont les hiérarchies qui permettent d'obtenir une carte mentale des données plutôt qu'une carte

physique. Nous examinons quelles sont ces techniques qui permettent d'obtenir ces hiérarchies que Camargo [Camargo et González, 2009] nomme « techniques de résumé ».

Ce chapitre s'articule de la façon suivante. Nous présentons au paragraphe 2.1 le « modèle de référence de la visualisation » qui décrit le processus d'élaboration d'une visualisation. Suit au paragraphe 2.2 la présentation d'un processus de cartographie sémantique et de ses espaces informationnels qui consiste en une adaptation du modèle de référence. Ce processus de cartographie sémantique sera notre point de départ pour l'élaboration du processus de visualisation que nous présentons au paragraphe 2.6.1. Afin de préciser notre processus, nous commençons par explorer au paragraphe 2.3 les taxonomies de la visualisation d'information. Ensuite, nous remarquons qu'une étape importante de cette production de visualisations est la structuration de la base documentaire. Au paragraphe 2.4, nous examinons quelles sont ces structurations possibles et quelles sont celles couramment visualisées. Pour aller plus loin, l'annexe A détaille les visualisations existantes en se focalisant sur celles de données multimédia de type image et vidéo. Le paragraphe 2.5 résume cette annexe pour en déduire une liste des représentations envisageables.

Ce chapitre se termine au paragraphe 2.6 par une synthèse présentant notre processus de visualisation qui met en évidence les besoins de structuration. Nous finissons en présentant comment notre processus structure les réalisations des chapitres suivants.

2.1 Généralités sur les visualisations

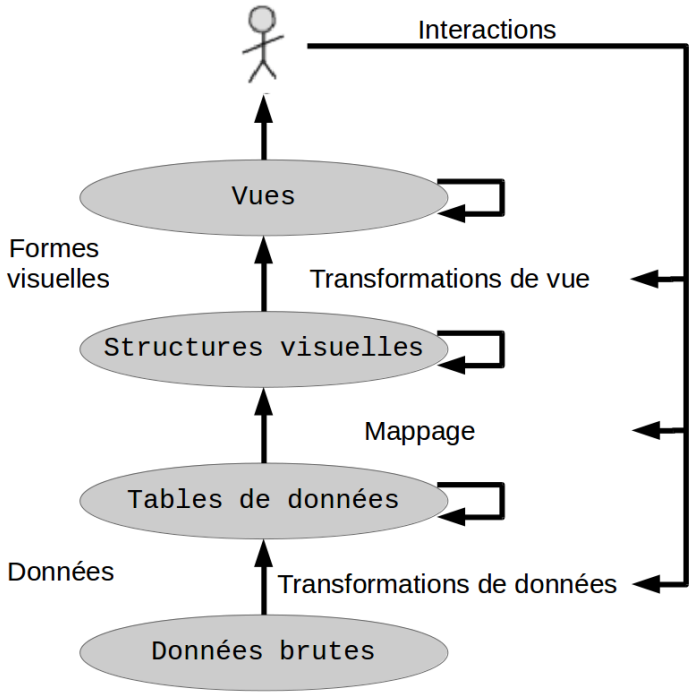


FIGURE 2.1 – Modèle de référence de la visualisation établi par Card, Mackinlay et Shneiderman en 1999.

Pfitzner [Pfitzner *et al.*, 2003] a mis en évidence cinq facteurs fondamentaux à prendre en compte pour la réalisation d'une application de visualisation d'information : les données, la

tâche, l'interaction, le niveau d'expertise de l'utilisateur et le contexte d'utilisation. En outre, la visualisation d'information dispose d'un modèle de référence élaboré par Card, Makinley et Schneiderman en 1999 (figure 2.1) Ce modèle est repris dans plusieurs livres comme celui sur la visualisation d'information écrit ultérieurement par Card [Card, 2007].

Le modèle part des données brutes en leur faisant subir un ensemble de transformations pour aboutir à des tables de données. Ces transformations peuvent consister en l'extraction de différents descripteurs. Des opérations de mappage permettent ensuite d'obtenir des structures visuelles. Ce peut être par exemple une projection dans un espace 2D. Des transformations permettent finalement d'obtenir la vue proposée à l'utilisateur. Ces transformations peuvent être par exemple la déformation de l'espace 2D via l'adjonction d'un zoom. Dans un deuxième temps, l'utilisateur final peut interagir avec tous les niveaux du modèle. Il peut modifier la vue par exemple dans le cas d'un zoom en déplaçant le centre d'intérêt du focus. Il peut modifier la structure visuelle en demandant à avoir un autre mappage comme par exemple, passer d'une vue géographique 2D de type carte routière à une vue permettant des déplacements en 3D. Il pourrait aussi modifier la table de données en demandant l'extraction de nouveaux descripteurs. Ce modèle de référence peut permettre d'analyser et décrire toutes les visualisations existantes quelles que soient leurs variabilités apparentes. Cependant, pour choisir une visualisation adaptée au besoin, il reste à savoir évaluer la qualité de ces visualisations et à prendre en compte la satisfaction de l'utilisateur.

2.2 Un processus de cartographie sémantique

Dans le cas de la visualisation des connaissances, Tricot [Tricot, 2006] propose un processus de cartographie sémantique avec plusieurs espaces informationnels : brut, structuré, représenté et visualisé. La construction de la visualisation et son processus d'interaction avec l'utilisateur sont analogues à ceux du modèle de référence de la visualisation d'information.

Dans la figure 2.2, l'espace informationnel brut est composé des données immédiatement disponibles qui en général sont non structurées. À cet espace informationnel brut, sont appliquées des opérations de structuration, comme l'extraction de descripteurs, pour obtenir un espace informationnel structuré. Cette structure peut ensuite être adaptée par des méthodes de projection, classification.... Nous explorerons l'état de l'art de ces différentes méthodes au chapitre 3. L'opération suivante consiste à représenter les données pour obtenir l'espace informationnel représenté. En général, la structure des données conditionne les représentations possibles. Par exemple, l'affichage d'images structurées en liste unidimensionnelle est souvent représenté en pellicule 2D ou en grille 2D. À cet espace représenté, sont ensuite appliquées des tâches de visualisation pour arriver à l'espace visualisé par l'utilisateur. Ces tâches peuvent être par exemple l'ajout d'un zoom. Finalement, l'utilisateur peut interagir sur chacun des espaces. Il peut modifier l'espace visualisé en déplaçant par exemple le centre dans le cas d'un zoom. Il peut modifier l'espace représenté en passant par exemple d'une représentation en pellicule à une grille. Il peut modifier la structure en changeant par exemple sa dimension. Il peut aussi modifier l'espace brut en ajoutant, supprimant ou modifiant les données. Ceci correspond couramment à l'utilisation de filtres.

Dans notre contexte, la dénomination de la cartographie sémantique (en particulier celle d'espace structuré) nous semble intéressante pour notre approche car adaptable à nos bases documentaires de type multimédia qui n'ont pas de structure intrinsèque. La première étape consiste donc bien à extraire des descripteurs. Que ceux-ci soient numériques, ordinaux,

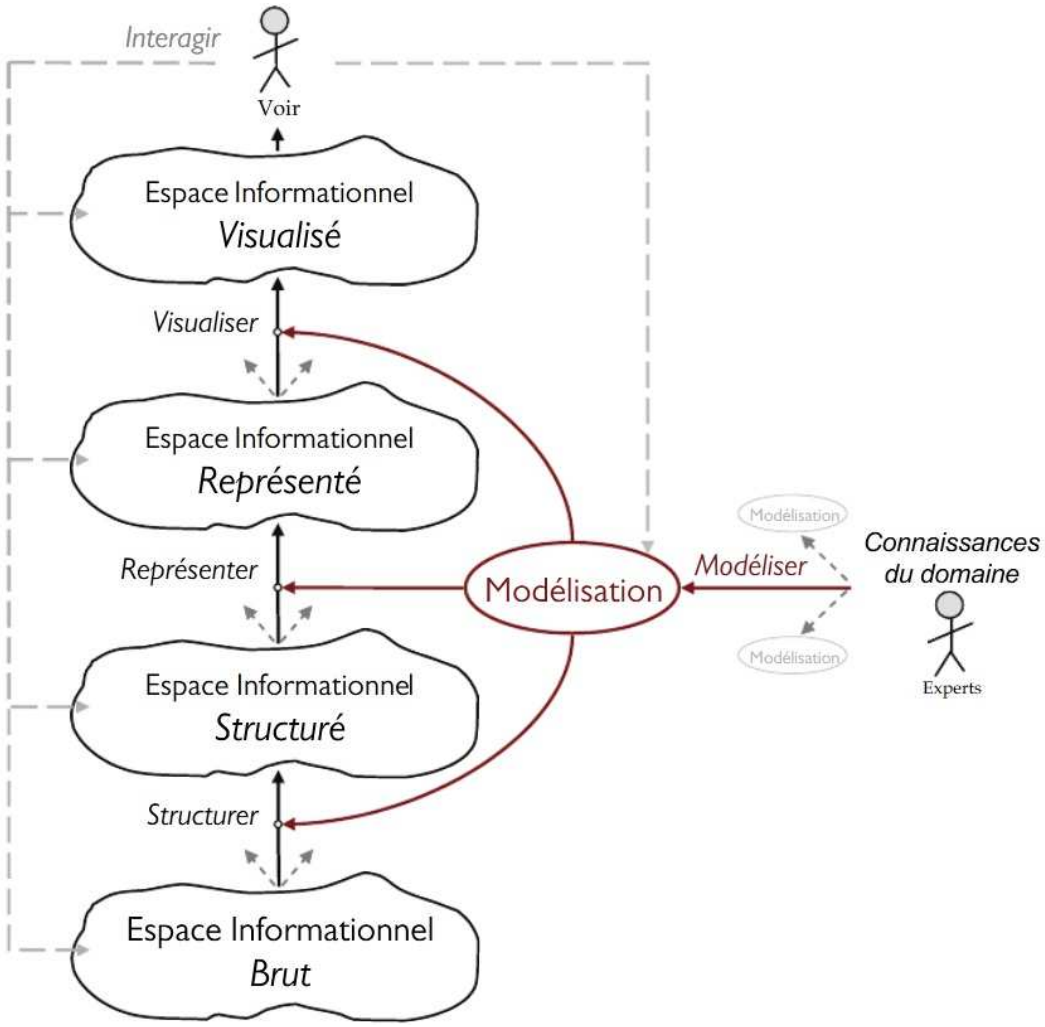


FIGURE 2.2 – Le processus de cartographie sémantique [Tricot, 2006].

cardinaux ou binaires, ce sont eux qui apportent une structure à la base. Ensuite tout le processus décrit précédemment peut être déroulé.

La base documentaire brute étant donnée, il reste à définir les trois autres espaces. Nous commençons par la visualisation qui doit répondre aux besoins des utilisateurs. Nous poursuivons par la structuration que nous pouvons apporter au données brutes et nous finissons par la représentation qui doit être la passerelle entre la structuration des données et la visualisation voulue.

2.3 Classification des visualisations

Shneiderman [Shneiderman, 1996] a mis en place la taxonomie Type by Task Taxonomy (TTT). Elle est certainement celle qui a le plus influencé la visualisation d'information [Chen, 2010]. Cette taxonomie est basée sur le type de données représentées et sur les tâches effectuées par l'utilisateur. Elle distingue 7 types de données : 1D, 2D, 3D, temporelles, multidimensionnelles, hiérarchiques et relationnelles. Et 7 tâches : avoir une vue d'ensemble,

zoomer, filtrer, obtenir des détails, lier les représentations, disposer d'un historique des actions réalisées et exporter une partie des informations vers d'autres applications.

Une tâche complexe comme par exemple, « Focus and Context » peut être décrite comme une combinaison des tâches « avoir une vue d'ensemble », « zoomer » et « lier les deux représentations » [Jaeschke *et al.*, 2005]. Cockburn [Cockburn *et al.*, 2009] compare trois tâches complexes : Vue d'ensemble + Détails, Zoom et Vue d'ensemble + Zoom en notant qu'aucune des trois ne se distingue nettement.

Pour ce qui concerne les données, de nombreuses taxonomies basées sur le TTT ont été proposées. Bruley [Bruley et Genoud, ntes] a séparé les données 1D, 2D et 3D en deux catégories selon un point de vue spatial d'une part, et un point de vue non structuré d'autre part. Tory [Tory et Möller, 2002] a quant à lui proposé une séparation entre des données continues et des données discrètes. Mais comme le précise Jaeschke [Jaeschke *et al.*, 2005], une quantité de taxonomie ont été proposées en extension de la TTT, mais aucune n'a jamais été aussi largement adoptée que le travail de Shneiderman. Cependant, Shneiderman considérait sa classification incomplète et prévoyait que les applications à venir allaient requérir des structures de données nouvelles et spécialisées.

Nous pouvons donc retenir comme classification des visualisations le résultat des 7 tâches de la visualisation présentées dans la figure 2.3.

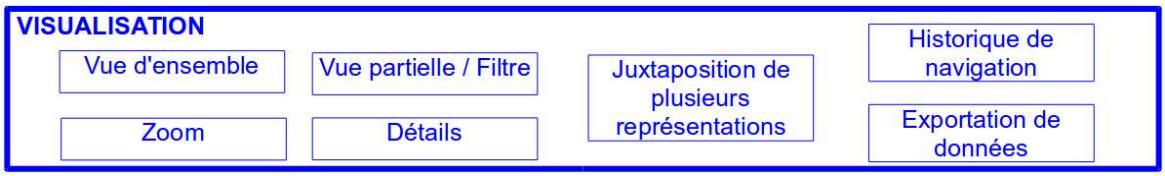


FIGURE 2.3 – La classification des visualisations correspondant au résultat des 7 tâches de la taxonomie TTT [Shneiderman, 1996].

2.4 Des données structurées

A ce point, nous avons :

- 7 types de données avec des subdivisions possibles entre le continu et le discret, entre le structuré physiquement de manière spatiale et le non structuré...
- un ensemble E composé d'une base documentaire des données brutes sans structure intrinsèque ;
- une opération de structuration qui transforme l'« espace brut » en un « espace structuré ».

Proposons comme définition de l'action de « **structurer** » : « créer une **relation** impliquant l'**ensemble** E ». Cette relation peut être :

- externe sur l'ensemble $E \times F$ avec $F \neq E$. Par exemple, on peut avoir une relation sur l'ensemble $E \times \mathbb{R}$ qui affecte à chaque élément de E une valeur réelle.
- interne sur l'ensemble $E \times E \times \cdots \times E$. Par exemple un graphe qui relie entre eux certains éléments de E est une relation sur l'ensemble $E^2 = E \times E$.
- interne/externe. Par exemple un graphe pondéré correspond à une relation sur l'ensemble $E^2 \times \mathbb{R}$.

Cette définition de la structuration est importante. Elle va nous permettre de formaliser et classifier ces structures. Pour ce travail, les travaux de Tricot [Tricot, 2006] sont un bon support.

Pour lister les paradigmes de représentation, Tricot ne retient aucune taxonomie existante, même s'il se base partiellement sur la TTT. Il distingue deux grandes classes de structures : celles qu'il qualifie d'« orientées valeurs » et celles qu'il qualifie d'« orientées relations ». Notons tout de suite qu'il y a ambiguïté entre le terme « relation » utilisé par Tricot et la définition mathématique que nous utilisons. Tricot utilise le terme général « relation » pour qualifier ce qui est uniquement le cas particulier des « relations binaires internes ». Dans toute la suite, nous nommerons structure « orientée liaisons » ce que Tricot qualifie de structure « orientée relations ».

Avec notre définition de l'action de « structurer », les structures orientées valeurs sont des relations binaires externes $E \times F$. Elles font intervenir un espace tiers F. Tricot les classe ensuite selon leur dimension. Cette notion de dimension est dans notre définition, la dimension de l'espace F. Nous pouvons voir que d'autres sous-classes apparaissent suivant les propriétés de la relation et de F. Par exemple, $F = \mathbb{R}$ a la puissance du continu, $F = \mathbb{N}$ est discret, la relation peut être injective sur F, bijective...

Lorsque Tricot évoque les relations binaires valuées, il s'agit de relations définies sur $E^2 \times \mathbb{R}$. Et pour ce qui est des hiérarchies définies comme étant un graphe orienté acyclique, il ne s'agit que d'un cas particulier des structures orientées liaisons.

Nous établissons une classification des représentations basée sur celle de Tricot et formalisée grâce à la définition de la structuration que nous avons proposé. D'où en résumé, la classification suivante en deux grandes familles avec leurs principales possibilités :

1. Les structures orientées valeurs qui sont les applications de E dans F
   (a) $dim(F) = 1$
      i. $F = \mathbb{R}$: un placement unidimensionnel (par exemple une chronologie)
      ii. $F = \{0; 1\}$: une classification binaire. Exemple : présent/absent
      iii. $F = \{a_1; a_2; \cdots; a_k\}$: une classification en k classes distinctes
      iv. $F = \mathbb{N}$: un ordonnancement
      v. $F = \frac{\mathbb{Z}}{n\mathbb{Z}}$: une liste circulaire
   (b) $dim(F) = 2$ avec principalement $F = \mathbb{R}^2$: un placement 2D
   (c) $dim(F) = 3$ avec principalement $F = \mathbb{R}^3$: un placement 3D
   (d) $dim(F) = n$ avec principalement $F = \mathbb{R}^n$: un placement multidimensionnel
2. Les structures orientées liaisons qui sont les applications de $E \times E$ dans F avec principalement :
   (a) $F = \{0; 1\}$: un graphe non orienté
   (b) $F = \mathbb{R}$: un graphe pondéré
   (c) $F = \{-1; 0; 1\}$: un graphe orienté
   (d) $F = \{-1; 0; 1\}$ **sans aucun cycle** : une hiérarchie

La figure 2.4 condense le contenu de cette classification. Afin de l'illustrer, examinons en détail un exemple de visualisation qui utilise une technique MDS « Multi Dimensional Scaling » itérative complétée par une visualisation en grille 2D [Liu *et al.*, 2004]. Il s'agit d'une visualisation de type Fisheye sur une représentation en grille illustrée dans la figure 2.5. La

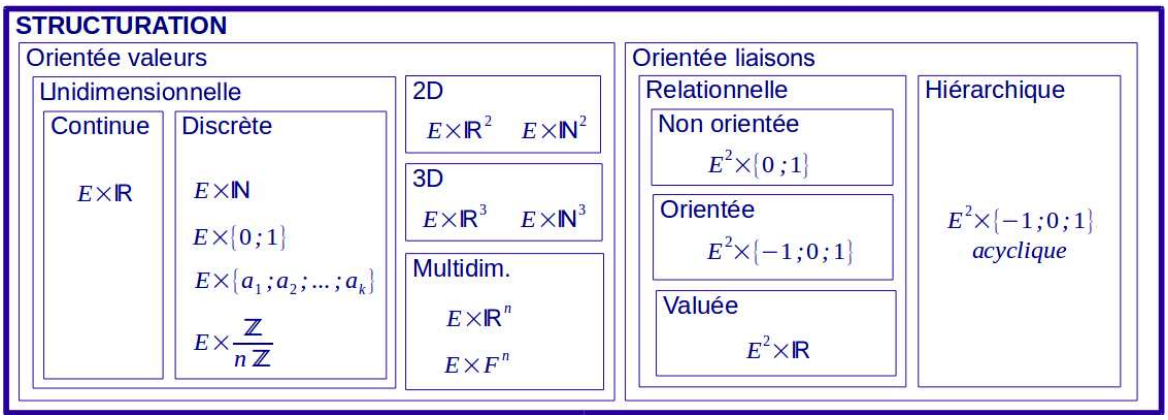


FIGURE 2.4 – Notre classification de la structuration d'un ensemble de données E.

FIGURE 2.5 – Visualisation de type Fisheye sur une grille [Liu *et al.*, 2004].

grille donnée en exemple dans la sous-figure de droite est remplie à partir d'une représentation bidimensionnelle représentée dans la sous-figure de gauche. Le processus, résumé par la figure 2.6, respecte le modèle de référence de la visualisation donné dans la figure 2.1. Il respecte aussi le passage par les différents espaces informationnels du processus de cartographie sémantique :

- l'espace brut est constitué de la base documentaire brute ;
- les différents espaces structurés sont successivement obtenus via :
  - une relation applicative $E \times \mathbb{R}^n$ obtenue par l'extraction des descripteurs ;
  - une relation applicative $E^2 \times \mathbb{R}$ obtenue par construction d'une matrice de distance euclidienne calculée sur les descripteurs ;
  - une relation applicative $E \times \mathbb{R}^2$ obtenue en appliquant la technique « projective » MDS qui, de manière itérative, place les points dans le plan 2D de façon à minimiser l'erreur quadratique entre les distances des points dans la plan 2D et les distances entre les images dans l'espace initial ;
  - une relation applicative $E \times \mathbb{N}^2$ injective obtenue par un algorithme de structuration en grille qui de façon dynamique ajuste les chevauchements jusqu'à obtenir une grille sans superposition des images ;
- l'espace représenté est obtenu en créant une représentation visuelle de la structure. Dans ce cas, on utilise la représentation naturelle de $E \times \mathbb{N}^2$: une grille 2D plane ;

- la visualisation proposée par Liu de cette grille est un « Focus and Detail » de type Fish-eye.

Et pour finir, les interactions offertes à l'utilisateur sont uniquement au niveau de l'espace visualisé, en permettant le déplacement du centre du fish-eye dans la grille.

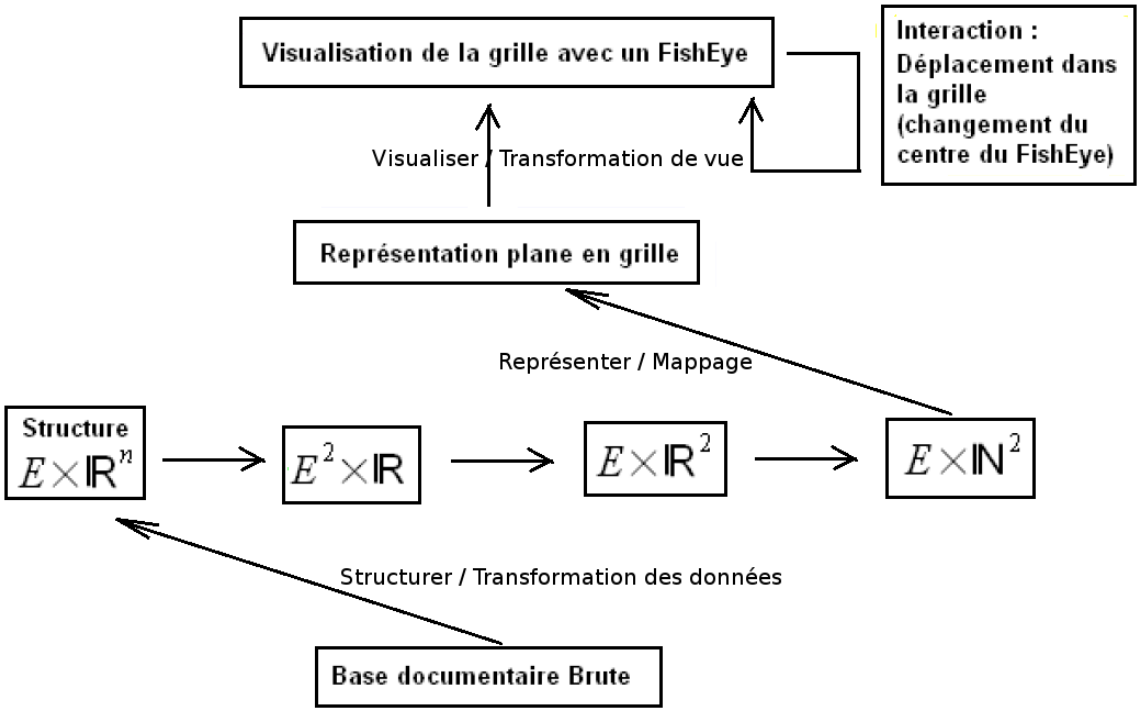


FIGURE 2.6 – Exemple de processus de cartographie décrit par Liu [Liu *et al.*, 2004].

2.5 Une liste des représentations envisageables

A ce point, avec la figure 2.2, nous avons un processus de visualisation à 4 couches (l'espace brut, la structuration, la représentation et la visualisation). Avec les figures 2.3 et 2.4, nous avons une classification de la visualisation et de la structuration. Il nous manque une classification ou plutôt une liste des représentations qui correspondent à ces structurations et visualisations. Pour la mettre au point, nous nous basons sur les visualisations existantes et surtout sur les représentations qu'elles utilisent.

En examinant notre classification des structures donnée dans la figure 2.4 et les visualisations existantes présentées en détail dans l'annexe A, nous pouvons établir le bilan suivant.

Les structures orientées valeurs continues ne sont que très peu représentées et visualisées. Elles sont surtout utilisées comme intermédiaire avant l'obtention de structures discrètes qui sont couramment représentées sous forme de pellicule 2D ou 3D, carrousel, grille. Les structures bidimensionnelles sont principalement représentées sous forme placements 2D et de grilles comme dans l'exemple de visualisation de type Fisheye présenté au paragraphe précédent (figure 2.5). Les structures de plus grande dimensionnalité peuvent par exemple être représentées sous forme de Placement 3D ou Grilles sphériques.

Pour ce qui concerne les structures orientées liaisons que sont les graphes et les hiérarchies,

les représentations sont principalement des placements en graphe développé ou radial et des remplissages rectangulaires. Il existe aussi des placements sphériques.

Pour illustrer ces hiérarchies avec placement rectangulaire, Gomi [Gomi *et al.*, 2008] propose une exploration hiérarchique de bases d'images utilisant une technique de remplissage rectangulaire. Le zoom peut porter sur chacun des niveaux de la hiérarchie comme illustré avec les 3 niveaux de la figure 2.7. La figure 2.8 montre trois niveaux de zoom sur une base réelle. Au troisième niveau de zoom, seul le sous-arbre encadré en rouge est visualisé. Le reste de l'arbre n'est plus affiché.

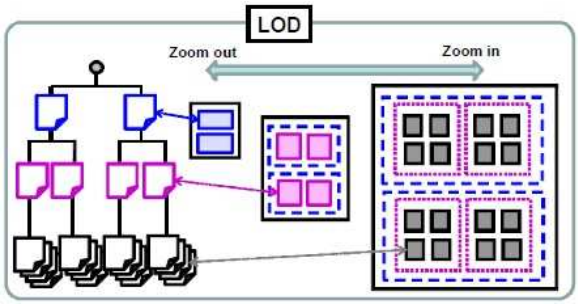


FIGURE 2.7 – Un exemple de base hiérarchique équilibrée à trois niveaux avec ses trois zooms correspondants [Gomi *et al.*, 2008].

FIGURE 2.8 – Un exemple du papier [Gomi *et al.*, 2008] avec trois niveaux de zoom sur une base hiérarchique réelle.

En conclusion, les visualisations existantes sont nombreuses. Nous pouvons en déduire que les représentations envisageables sont elles aussi nombreuses et que nous ne pouvons pas en donner une liste exhaustive ou une classification. Mais nous pouvons donner une liste correspondant aux classifications des structurations et visualisations des figures 2.3 et 2.4. Cette liste des représentations existantes est résumée dans la figure 2.9. Les points de suspension sont là pour indiquer que cette liste n'est pas limitative.

FIGURE 2.9 – Notre liste non exhaustive des représentations existantes.

2.6 Synthèse, bilan et objectifs

2.6.1 Synthèse : un processus pour l'élaboration d'une visualisation

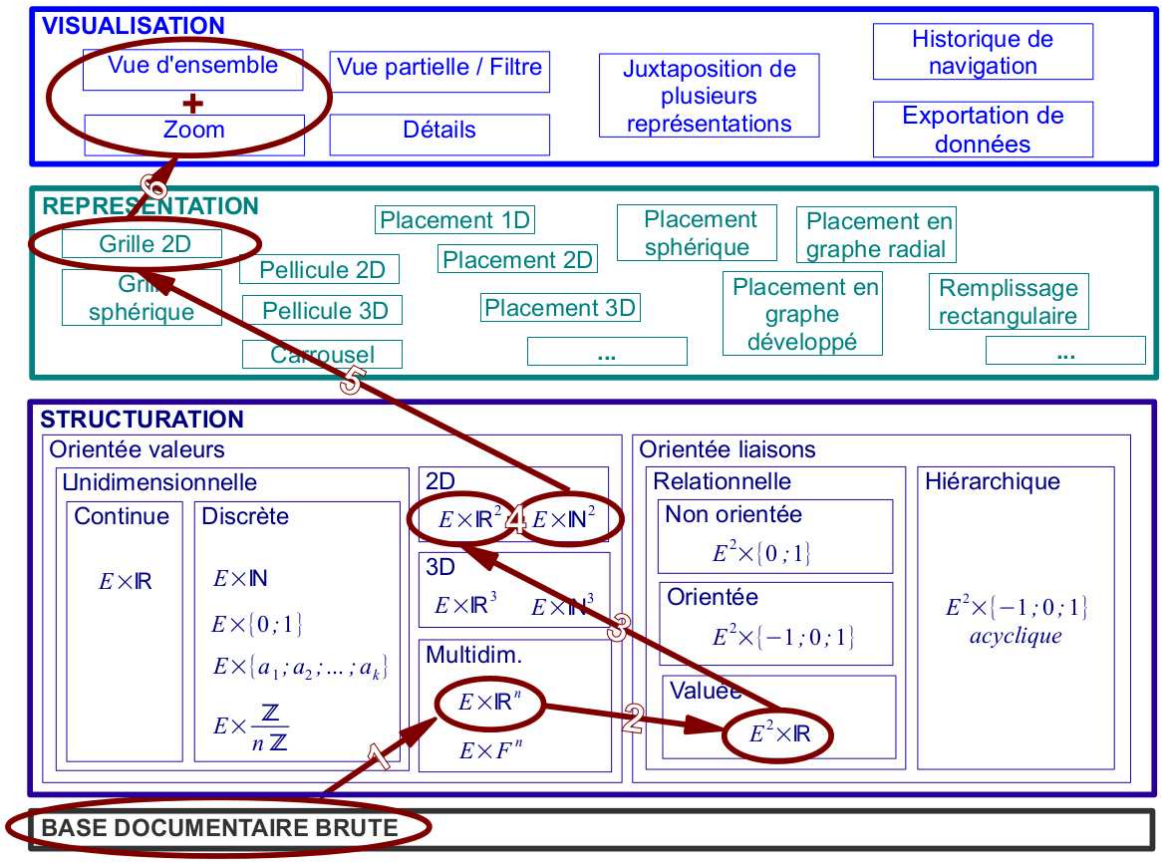


FIGURE 2.10 – Le modèle de visualisation adapté à une base documentaire multimédia. En rouge, figure le processus de visualisation proposé par Liu [Liu *et al.*, 2004] et présenté au paragraphe 2.4.

Le modèle de référence de la visualisation présenté au paragraphe 2.1 décrit les transformations successives qui permettent d'obtenir une visualisation à partir des données brutes. En respectant ce modèle, la figure 2.10 résume le processus d'élaboration d'une visualisation d'une base de données de type multimédia avec ses 4 niveaux. Le premier encadré (en noir) est la base documentaire brute sans structure inhérente. Le deuxième (en bleu foncé), repris de la figure 2.4, résume les structures des données présentées au paragraphe 2.4. Le troisième

niveau (en turquoise), repris de la figure 2.9, recense les représentations citées au paragraphe 2.5. Le quatrième bloc, déjà vu avec la figure 2.3, reprend les résultats des 7 tâches de la taxonomie **T**ype by **T**ask **T**axonomy décrite au paragraphe 2.3.

Le processus d'élaboration d'une visualisation correspond à un processus partant de la base documentaire brute pour aboutir à la visualisation. A titre d'exemple, figure en rouge, le processus de visualisation présenté au paragraphe 2.4. Les 6 étapes du processus, représentées par les 6 flèches rouges, sont « l'extraction de n descripteurs », « le calcul de la matrice de distance », « une projection bidimensionnelle », « une structuration en grille à l'aide d'un algorithme itératif », « une représentation en grille 2D » et « une visualisation offrant une vue d'ensemble complétée par un zoom ».

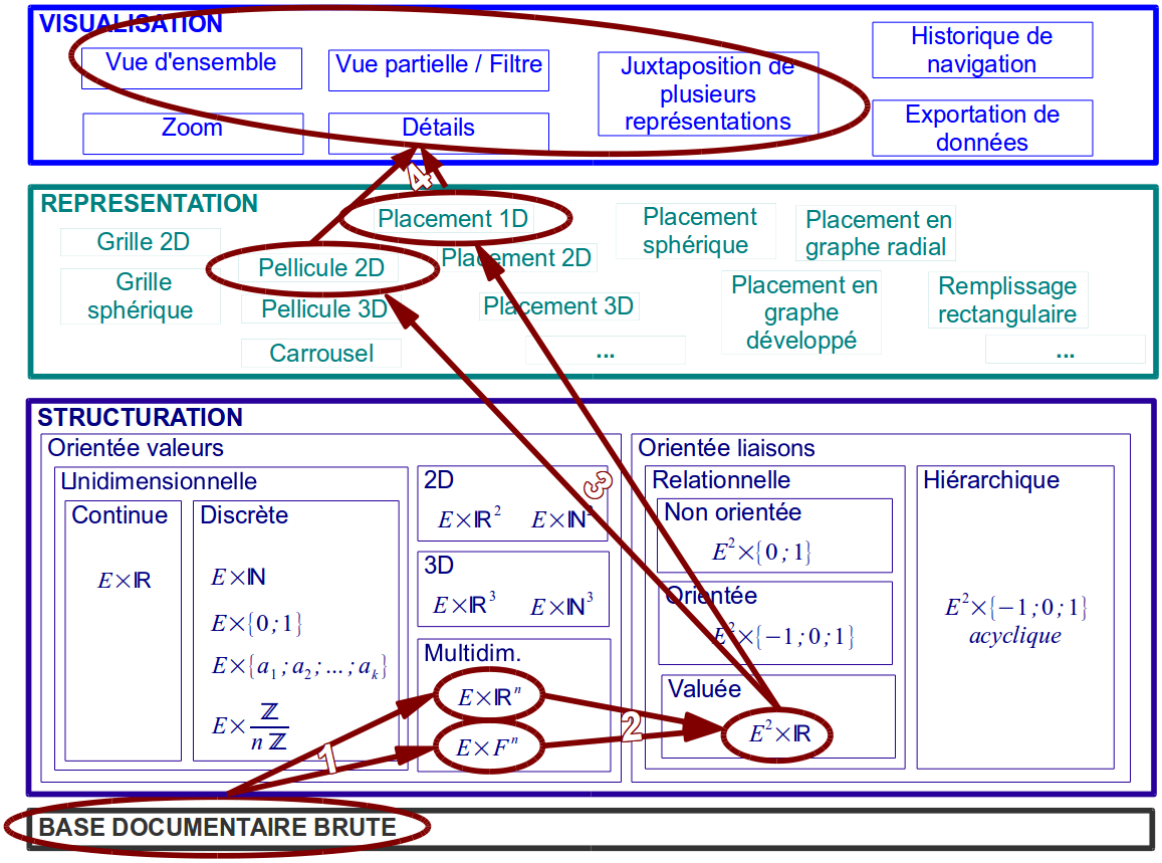


FIGURE 2.11 – Le modèle de visualisation adapté à la méthode de fusion de descripteurs et au prototype de visualisation implémenté au chapitre 4. En rouge, figure le processus complet avec l'extraction et la fusion des descripteurs suivies de la représentation et de la visualisation du prototype que nous avons réalisé.

Les processus de structuration commencent souvent comme celui présenté ci-dessus : après une extraction de n descripteurs et la construction d'une matrice de similarité ou de distance, une projection est effectuée. Cependant, à l'intérieur de ce schéma, beaucoup d'autres cheminements sont possibles.

Dans le chapitre 4, nous développons une technique originale de fusion de métriques à l'aide des coefficients de corrélation de rang. Son processus est synthétisé dans la figure 2.11. La première étape consiste en l'extraction de descripteurs numériques ($E \times \mathbb{R}^n$) et de type

métadonnée ($E \times F^n$). La seconde étape est la fusion pour aboutir à une unique métrique ($E^2 \times \mathbb{R}$). Les étapes suivantes résument le processus d'élaboration du prototype de logiciel d'exploration de base de données vidéos que nous avons réalisé.

Dans le chapitre 5, nous nous intéressons au Clustering Spectral dont le processus de structuration est en trois étapes. Après une étape préliminaire d'extraction des descripteurs ($E \times \mathbb{R}^n$), la figure 2.12 présente les trois étapes du Clustering Spectral : la construction du graphe de similarité ($E^2 \times \mathbb{R}$), la projection dans un espace spectral de plus faible dimension k ($E \times \mathbb{R}^k$ avec $k < n$) et le partitionnement des données dans l'espace spectral ($E \times \{a_1; a_2; \cdots a_k\}$) en k classes $a_1, a_2, \cdots, a_k$. Les couches de représentation et de visualisation ne sont détaillées car elles dépendront du type d'application utilisée pour visualiser ce partitionnement.

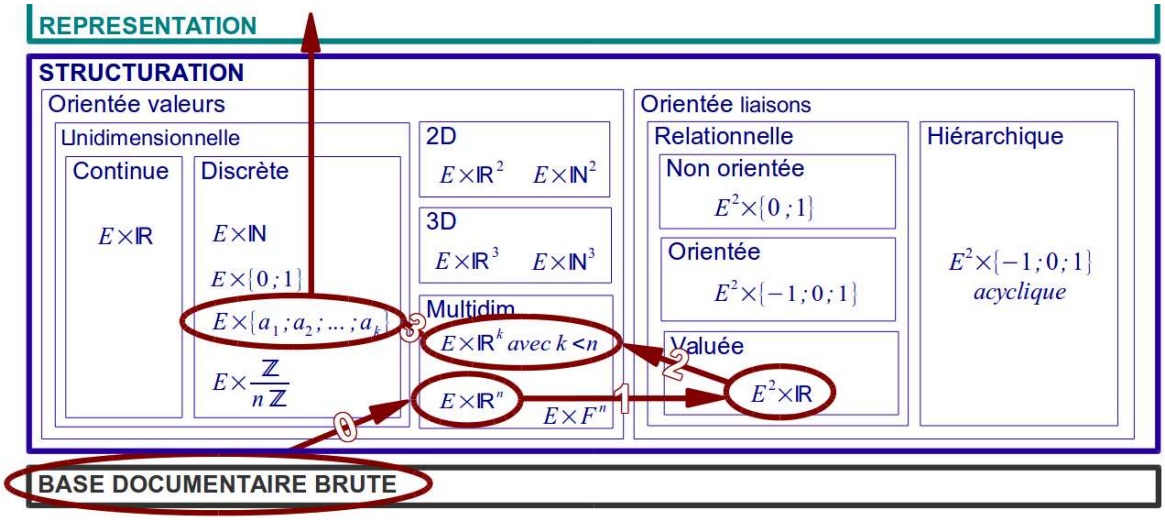


FIGURE 2.12 – La structuration du modèle de visualisation adapté au Clustering Spectral. En rouge, figure le processus de structuration avec l'extraction des descripteurs et les trois étapes du Clustering Spectral.

Pour ce qui concerne les deux dernières étapes du processus, le choix de la représentation est issu de l'expérience. Les représentations sont liées aux structures car, comme détaillé dans l'annexe A, chaque structure possède quelques représentations qui lui sont naturellement appliquées. Au paragraphe 2.1, nous avons vu que le choix d'une visualisation de données de type multimédia doit tenir compte du contexte d'utilisation, du niveau d'expertise de l'utilisateur, de la tâche et des interactions souhaitées. Il est donc utile d'effectuer des expériences de validations avec utilisateur et d'effectuer des enquêtes spécifiques pour choisir une visualisation adaptée. Ce type de travaux est plus spécifique de la communauté IHM et dépasse le cadre de cette étude.

2.6.2 Bilan et objectifs

A ce point, nous constatons que les visualisations sont bien décrites dans la littérature. Le schéma avec les différentes couches que nous avons détaillées dans la figure 2.10 est clair. Pour produire une application de visualisation d'une base multimédia, il suffit alors de choisir le bon processus de visualisation. C'est à dire, choisir la visualisation adaptée, déduire la représentation correspondante, puis à partir des spécificités de la base documentaire brute,

élaborer un processus de structuration adéquat, de manière analogue à celui de l'exemple présenté au paragraphe 2.4. Mais ceci n'est qu'un travail de développement logiciel en relation forte avec les attentes des utilisateurs. Tous ces aspects directement liés à la visualisation concernent l'ergonomie IHM que nous avons décidé de ne pas approfondir plus. Cependant, nous avons mis en évidence que des besoins de recherche existent sur les techniques de structuration des données. Nous avons choisi de nous intéresser à ces techniques qui font partie du Machine Learning et le travail effectué dans cette thèse s'articule donc autour de ces besoins.

Chapitre 3

L'état de l'art

Résumé : Dans ce chapitre nous présentons l'état de l'art sur la structuration des données. Nous commençons par décrire les spécificités des données et comment produire des mesures de proximité. Puis nous examinons les techniques de structuration par projection qui permettent de diminuer la dimension des données. Ensuite nous nous intéressons aux techniques de structuration par classification qui permettent de regrouper ensemble les données qui se ressemblent. Nous présentons ce que ces classifications produisent : des classes et des partitions souvent organisées en hiérarchies. Nous examinons comment évaluer la qualité de ces classifications. Nous nous intéressons aussi à l'ajout de connaissance dans la classification en examinant les méthodes automatiques, supervisées et semi-supervisées. Et pour finir nous détaillons une technique qui nous intéressera dans nos réalisations : le Clustering Spectral et la théorie spectrale des graphes qu'il met en œuvre.

Nous avons vu au chapitre 2 qu'une étape importante du processus de visualisation est la structuration des données. Cette structuration vise à retravailler les données de façon à ce qu'elles soient facilement représentables sous la forme désirée. Par exemple, si l'on souhaite une visualisation en grille, il faut restructurer les données E sous la forme $E \times \mathbb{N}^2$.

En préalable à la structuration des données, il faut examiner de quel type sont les données et savoir produire des mesures de ressemblance sur celles-ci :

- Les données peuvent être représentées par des variables qualitatives ou quantitatives.
- Les mesures de ressemblances peuvent être des distances ou simplement des dissimilarités.

Selon le type de variables, nous pouvons produire différents types de distances ou d'indices de similarité. C'est ce que nous examinons au paragraphe 3.1.

La première étape de structuration des données consiste souvent en l'extraction de n variables. Mais la structure multidimensionnelle ainsi obtenue n'est que très peu directement représentée. Les données sont plutôt remaniées pour aboutir à l'une des deux grandes familles de structures représentables : les structures orientées valeurs et les structures orientées liaisons. Ceci nous amène donc à étudier quelles sont les méthodes permettant d'obtenir ces deux familles de structures.

Camargo [Camargo et González, 2009] nomme « techniques de visualisation » toutes les méthodes qui permettent de projeter les données dans un espace visualisable. Ces méthodes

de projection permettent d'obtenir une structure orientée valeurs de faible dimension. Les données sont alors toutes visualisables sur une carte bi ou tridimensionnelle. Au paragraphe 3.2, nous étudions les principales techniques de projection. Les plus anciennes sont éprouvées et ont été proposées au siècle dernier. D'autres sont beaucoup plus sophistiquées et récentes.

Au paragraphe 3.3, nous nous intéressons aux techniques de classification car elles produisent des structures orientées liaisons, souvent de type hiérarchique, ce qui permet de visualiser des résumés de la base, c'est-à-dire de naviguer dans des extraits représentatifs de la base. Nous commençons ce paragraphe par une présentation des classes, partitions et hiérarchies. Nous décrivons aussi les multiples indicateurs de séparabilité et d'homogénéité qui permettent de mesurer la qualité des classes et partitions obtenues. Les méthodes de classification fournissent parfois des partitions organisées en hiérarchies. Nous détaillons ces hiérarchies pour comprendre ce qu'elles apportent comme structuration supplémentaire par rapport à un simple partitionnement. Nous présentons ensuite l'ajout de connaissance au travers des techniques de classification automatique, supervisée et semi-supervisée. Nous détaillons les principales méthodes de classification existantes ainsi que celles utilisées dans nos réalisations.

Au paragraphe 3.4, nous finissons par une présentation détaillée de la méthode du Clustering Spectral et de la théorie spectrale des graphes qui est à la base de cette méthode.

3.1 Données et mesures

Au préalable à la production de mesures de proximité entre les objets, il convient de s'intéresser au type de variables qui les décrivent.

Soit $E = \{x_1; x_2; \cdots; x_n\}$ un ensemble fini de n objets. De ces objets, nous pouvons extraire un nombre fini N de variables qui décrivent les objets. On parle alors de descripteur de dimension N. Chaque dimension peut être de deux natures différentes : les variables qualitatives et les variables quantitatives.

3.1.1 Variables qualitatives

Ce sont des variables qui s'expriment en modalités. Elles sont représentées par des qualités, comme par exemple le degré de satisfaction, la couleur des yeux ou le sexe d'un individu. Elles sont de deux sous types différents : les variables ordinales et les variables nominales. Les variables binaires sont des variables nominales particulières.

- Les variables qualitatives ordinales sont les variables qualitatives qui comportent un ordre. Ce peut être un degré de ressemblance à 5 modalités : « très ressemblant », « ressemblant », « neutre », « différent », « très différent ».
- Les variables qualitatives nominales sont des variables qui prennent un nombre fini de modalités et qui sont a priori non comparables (sans ordre). Ce peut être par exemple une profession, une couleur, une technique, un genre, un pays...
- Les variables qualitatives binaires sont les variables qualitatives nominales qui ne prennent que deux valeurs. Elles peuvent définir une présence/absence lorsqu'elles sont codées par 1 et 0. Lorsqu'elle sont codées autrement, on parle de variables dichotomiques. Par exemple, 1 et 2 pour le sexe du numéro INSEE.

3.1.2 Variables quantitatives

Ce sont des variables qui se rapportent à une quantité, comme par exemple le nombre d'enfants ou la taille d'un individu. Elles sont classées en 2 sous types différents.

- Les variables quantitatives continues sont des variables quantitatives telles qu'entre deux valeurs observées toutes les valeurs sont a priori observables. C'est le cas de la taille, du poids ou de la température qui sont des grandeurs numériques réelles.
- Les variables quantitatives discrètes sont par opposition aux variables quantitatives continues, les variables quantitatives qui sont assimilables à un ensemble de points à coordonnées entières. C'est le cas du nombre d'enfant ou de l'âge d'un individu. Les variables temporelles comme l'année sont des variables discrètes qui utilisent les unités de temps.

3.1.3 Similarités, dissimilarités et distances

De nombreuses mesures de distance et indices de similarités existent dans la littérature. Cette grande variété s'explique, entre autre, par les différentes natures des variables décrites précédemment. Ces mesures de proximité ont différentes propriétés et peuvent être classées selon celles qu'elles vérifient.

Une *similarité* est une application s de $E \times E$ vers $\mathbb{R}$, positive, symétrique et maximale :

$$\text{Positivité :} \quad \forall (x,y) \in E^2 \quad s(x,y) \geq 0 \tag{3.1}$$
$$\text{Symétrie :} \quad \forall (x,y) \in E^2 \quad s(x,y) = s(y,x) \tag{3.2}$$
$$\text{Maximale :} \quad \forall (x,y) \in E^2 \quad s(x,x) \geq s(x,y) \tag{3.3}$$

Une *similarité normalisée* est une similarité qui prend ses valeurs dans l'intervalle $[0,1]$. De manière assez courante, le terme similarité désigne une similarité normalisée.

Une *dissimilarité* est une application d de $E \times E$ vers $\mathbb{R}$, positive, symétrique et alternée :

$$\text{Positivité :} \quad \forall (x,y) \in E^2 \quad d(x,y) \geq 0 \tag{3.4}$$
$$\text{Symétrie :} \quad \forall (x,y) \in E^2 \quad d(x,y) = d(y,x) \tag{3.5}$$
$$\text{Alternée :} \quad \forall x \in E \quad d(x,x) = 0 \tag{3.6}$$

Une *dissimilarité propre* est une dissimilarité qui vérifie en plus la propriété de séparation :

$$\text{Séparation} : \forall (x,y) \in E^2 \quad d(x,y) = 0 \Leftrightarrow x = y \tag{3.7}$$

Une *distance* est une dissimilarité propre qui vérifie en plus l'inégalité triangulaire :

$$\text{Inégalité triangulaire} : \forall (x,y,z) \in E^3 \quad d(x,z) \leq d(x,y) + d(y,z) \tag{3.8}$$

Une distance est dite *ultramétrique* si elle vérifie la propriété d'ultramétrie :

$$\text{Ultramétrie} : \forall (x,y,z) \in E^3 \quad d(x,z) \leq \max\left(d(x,y); d(y,z)\right) \tag{3.9}$$

L'ultramétrie dote l'ensemble des objets d'une structure hiérarchique. Ceci permet d'effectuer facilement des regroupements hiérarchiques en classification automatique ascendante.

Les similarités (normalisées) s et les distances ou dissimilarités d peuvent être mises en correspondance par différentes relations comme par exemple :

$$d(x,y) = 1 - s(x,y) \tag{3.10}$$

$$s(x,y) = e^{\frac{-d^2(x,y)}{\sigma}} \quad \text{avec } \sigma \text{ une constante} \tag{3.11}$$

3.1.4 Les p-distances

Avec des variables quantitatives, les objets sont assimilables à des éléments $\mathbb{R}^N$. Nous pouvons calculer facilement des distances entre objets à l'aide des p-distances.

Considérons deux points de $\mathbb{R}^N$, $x = (x_i)_{1 \leq i \leq N}$ et $y = (y_i)_{1 \leq i \leq N}$.

La distance naturelle de notre monde euclidien est la distance euclidienne obtenue par généralisation du théorème de Pythagore à des dimensions supérieures.

- La *distance euclidienne* ou distance L_2 est :

$$d(x,y) = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2} \tag{3.12}$$

- La *distance de Manhattan* ou *distance city block* ou distance L_1 est :

$$d(x,y) = \sum_{i=1}^{N} |x_i - y_i| \tag{3.13}$$

- La *p-distance de Minkowski* ou distance L_P est :

$$d(x,y) = \sqrt[p]{\sum_{i=1}^{N} |x_i - y_i|^p} \tag{3.14}$$

- La *distance de Tchebychev* ou distance L_∞ est :

$$d(x,y) = \lim_{p \to \infty} \sqrt[p]{\sum_{i=1}^{N} |x_i - y_i|^p} = \max_{1 \leq i \leq N} |x_i - y_i| \tag{3.15}$$

3.1.5 La distance de Mahalanobis

La distance de Mahalanobis est utilisable avec des variables quantitatives. Elle prend en compte la variance et la corrélation des données. La distance de Mahalanobis entre deux points de $\mathbb{R}^N$, $x = (x_i)_{1 \leq i \leq N}$ et $y = (y_i)_{1 \leq i \leq N}$ est :

$$d(x,y) = \sqrt{(x-y)^T \Sigma^{-1} (x-y)} \tag{3.16}$$

où Σ est la matrice de covariance.

Si la matrice de covariance est la matrice identité, il s'agit de la distance euclidienne. C'est le cas des données normalisées par centrage-réduction.

Si la matrice de covariance est diagonale, il s'agit de la distance euclidienne normalisée :

$$d(x,y) = \sqrt{\sum_{i=1}^{N} \frac{(x_i - y_i)^2}{\sigma_i^2}} \tag{3.17}$$

où σ_i est l'écart type suivant la $i^{\text{ème}}$ composante.

3.1.6 Indice et distance de Jaccard

Si l'on considère deux ensembles A et B, l'indice de Jaccard entre les 2 ensembles est le rapport du cardinal de l'intersection sur celui de l'union :

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \tag{3.18}$$

La distance associée entre les 2 ensembles est :

$$d_J(A,B) = 1 - \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cup B| - |A \cap B|}{|A \cup B|} \tag{3.19}$$

Nous pouvons utiliser cet indice pour calculer la distance entre deux textes ou ensembles de mots. Par exemple, si

$$A = \{petit; oiseau; devenir; migrateur\}$$
$$B = \{petit; homme; devenir; ami; oiseau\}$$

alors

$$A \cap B = \{devenir; oiseau; petit\}$$
$$A \cup B = \{ami; devenir; homme; migrateur; oiseau; petit\}$$

et donc

$$J(A,B) = \frac{3}{6}$$

Si l'on considère les ensembles A et B inclus dans un ensemble E de cardinal n et si l'on considère tous les éléments de E comme des variables qualitatives binaires de type absence/présence, alors les ensembles A et B sont des objets caractérisés par n variables valant 0 ou 1. On peut en déduire que l'indice de Jaccard est aussi adapté pour mesurer la similarité entre des objets définis par un ensemble de variables qualitatives binaires.

Considérons $x = (x_i)_{1 \leq i \leq n}$ et $y = (y_i)_{1 \leq i \leq n}$ deux éléments de $\{0;1\}^n$ et les nombres :

$$N_{11} = \text{Card}\,\{i \text{ tels que } x_i = 1 \text{ et } y_i = 1\}$$
$$N_{10} = \text{Card}\,\{i \text{ tels que } x_i = 1 \text{ et } y_i = 0\}$$
$$N_{01} = \text{Card}\,\{i \text{ tels que } x_i = 0 \text{ et } y_i = 1\}$$
$$N_{00} = \text{Card}\,\{i \text{ tels que } x_i = 0 \text{ et } y_i = 0\}$$

On a alors $N_{11} + N_{01} + N_{10} + N_{00} = n$.

L'indice de Jaccard entre x et y est :

$$J(x,y) = \frac{N_{11}}{N_{01} + N_{10} + N_{11}} = \frac{N_{11}}{n - N_{00}} \tag{3.20}$$

La distance associée est alors :

$$d_J(x,y) = \frac{N_{01} + N_{10}}{N_{01} + N_{10} + N_{11}} = \frac{N_{01} + N_{10}}{n - N_{00}} \tag{3.21}$$

Avec l'exemple représenté dans le tableau 3.1 et l'équation 3.20, nous avons :

$$J(A,B) = \frac{N_{11}}{N_{01} + N_{10} + N_{11}} = \frac{3}{2+1+3} \tag{3.22}$$

	ami	devenir	homme	migrateur	oiseau	petit
A	0	1	0	1	1	1
B	1	1	1	0	1	1

TABLE 3.1 – Les ensembles $A = \{petit; oiseau; devenir; migrateur\}$ et $B = \{petit; homme; devenir; ami; oiseau\}$ mis sous forme de 2 objets à 6 variables binaires.

3.1.7 Autres indices de similarité

Comme décrit dans le livre [Brucker et Barthélemy, 2007], si l'on considère deux ensembles A et B, nous avons également les indices de similarité suivants où $|A|$ désigne le cardinal de A :

- Braun-Blanquet :

$$\frac{|A \cap B|}{\max\{|A|, |B|\}} \tag{3.23}$$

- Simpson :

$$\frac{|A \cap B|}{\min\{|A|, |B|\}} \tag{3.24}$$

- Ochiaï ou Driver et Krœber :

$$\frac{|A \cap B|}{\sqrt{|A| . |B|}} \tag{3.25}$$

- Czekanowsi ou Dice :

$$\frac{2|A \cap B|}{|A| + |B|} \tag{3.26}$$

Toutes ces similarités sont normalisées et avec la formule 3.10 nous pouvons obtenir la dissimilarité associée.

Cependant, cette liste n'est pas exhaustive. Il existe encore d'autres indices dont la plupart sont issus de l'écologie ou l'ethnologie [Brucker et Barthélemy, 2007].

3.2 Structuration par projection

Les variables et les mesures de dissimilarités étant décrites dans la paragraphe 3.1, nous avons les outils de base pour nous intéresser à la première famille de méthode de structuration des données : les projections. Elles nous permettent d'obtenir une représentation des données dans un espace de plus petite dimension. Ceci peut être, par exemple, une carte 2D des données.

Les différentes méthodes ont chacune leurs spécificités. Lorsque les données initiales contiennent des structures sous forme de variétés définies par connexité comme le SwissRoll 3D présenté dans la figure 3.2, certaines méthodes sont capables de déplier ces variétés contrairement à d'autres. La volonté de privilégier le respect des petites distances ou au contraire celui des grandes distances nous amène aussi à préférer certaines méthodes à d'autres.

Nous allons commencer par les plus anciennes méthodes qui sont les mieux connues pour finir par les plus récentes.

3.2.1 Analyse en Composante Principale (ACP)

L'ACP a été proposée en 1901 par Pearson [Pearson, 1901]. La méthode consiste en une projection linéaire sur le sous-espace engendré par les premiers vecteurs propres de la matrice de covariance. La figure 3.1 représente les composantes principales d'un nuage de points 2D. La figure 3.2 présente une répartition 3D en « Swiss Roll » et sa projection ACP en 2D. La perte d'information est dans les directions des composantes non principales. Cette méthode présente l'inconvénient de ne pas conserver la géométrie locale : comme présentés dans la figure 3.2, des points éloignés sur le « Swiss Roll » 3D peuvent se retrouver proches ou même confondus sur la projection 2D.

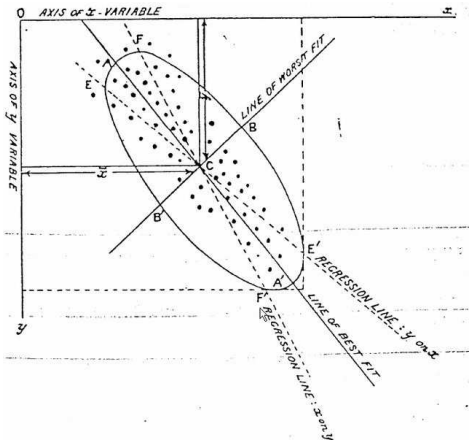


FIGURE 3.1 – Extrait de l'article de Pearson [Pearson, 1901].

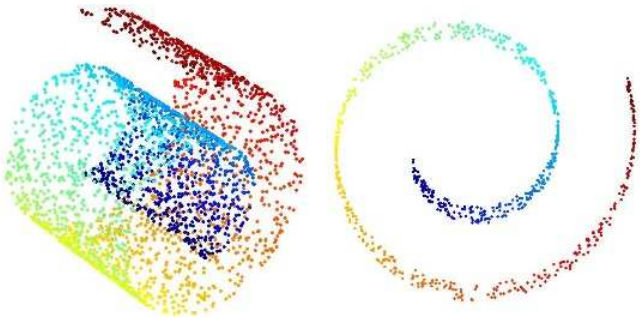


FIGURE 3.2 – Répartition 3D en « Swiss Roll » et sa projection ACP en 2D obtenue avec Matlab®.

3.2.2 Positionnement multidimensionnel (MDS)

Nommé aussi *Analyse en Coordonnées Principales*, le positionnement multidimensionnel [Borg et Groenen, 2005] est une dénomination générale pour un ensemble de méthodes dont la stratégie consiste à déterminer le sous espace projectif qui préserve au mieux les mesures

inter-points. C'est-à-dire un espace tel que deux objets semblables soient représentés par deux points proches et deux objets dissemblables par des points éloignés. Si l'on considère deux objets i et j avec d_{ij} la dissimilarité entre les deux objets dans l'espace de départ et δ_{ij} la dissimilarité entre les deux mêmes objets dans l'espace d'arrivée, on peut définir une fonction de coût à partir de l'erreur quadratique : $E = \sum_{i,j=1}^{n} (d_{ij} - \delta_{ij})^2$. Dans ce cadre le positionnement multidimensionnel consiste à minimiser cette fonction de coût.

Il existe plusieurs variations au MDS :

- Le *MDS classique* ou *MDS de Torgerson* décrit en 1952 ([Torgerson, 1952]) prend en entrée une dissimilarité quelconque entre les objets. Puis il calcule des coordonnées pour les objets dans un espace de plus petite dimension muni d'une distance euclidienne. Le problème est transcrit en un problème et un calcul purement algébriques : il consiste en une diagonalisation de matrice.
- Le *MDS métrique* est une généralisation du MDS classique par l'utilisation de différentes fonctions de coût qui sont minimisées par l'application d'un algorithme itératif. Les points sont positionnés puis déplacés itérativement de façon à minimiser la fonction de coût.
- Le *MDS non métrique* ne cherche pas une optimisation du respect des dissimilarités mais seulement un respect des rangs de proximité. Pour quatre objets i, j, k et l tels que $d_{ij} < d_{kl}$, l'algorithme cherche seulement à placer les objets dans l'espace d'arrivée de façon à ce que $\delta_{ij} < \delta_{kl}$.
- Le *MDS généralisé* est une extension du MDS métrique avec un espace d'arrivée qui n'est pas forcément euclidien.

Ces méthodes avec des algorithmes itératifs peuvent se présenter coûteuses en temps de calcul. Cependant en choisissant des fonctions de coût qui augmentent le poids des petites ou des grandes distances, nous pouvons obtenir des méthodes répondant à nos attentes.

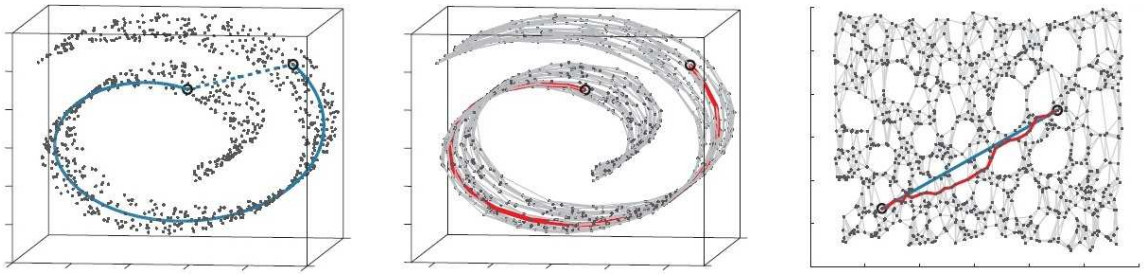
3.2.3 Isometric Mapping (Isomap)

Isomap a été décrite en 2000 [Tenenbaum *et al.*, 2000]. Il s'agit d'une adaptation du MDS classique qui cible une minimisation des distances géodésiques inter-points. Cette méthode procède en 3 étapes :

- Construction du graphe des k plus proches voisins (k-Isomap) ou des distances inférieures à un seuil ε (ε-Isomap). Le graphe est pondéré par les distances euclidiennes.
- Calcul des plus courts chemins (pondérés) dans ce graphe.
- Utilisation du MDS classique avec cette nouvelle matrice de distance pour obtenir les positions des objets dans l'espace d'arrivée.

La figure 3.3 illustre la technique Isomap avec un nuage de points en 3D projetés sur le plan 2D. Les sous-figures 3.3(a) et 3.3(b) représentent le nuage de points 3D qui est une répartition de type Swiss Roll. La sous-figure 3.3(c) représente les mêmes points projetés sur l'espace 2D. Sur les trois sous-figures sont entourés en noir deux points choisis arbitrairement. Figurent en trait pointillé bleu, la distance euclidienne de $\mathbb{R}^3$, en trait continu bleu la distance géodésique et en rouge le plus court chemin passant par les arêtes du graphe. La distance géodésique est la longueur du plus court chemin porté par la surface qui inclut les points du Swiss Roll 3D. On voit dans la figure 3.3(b) que la courbe rouge qui représente le plus court

chemin porté par le graphe de proximité, est une approximation de la courbe bleu de la figure 3.3(a). En utilisant cette approximation, la méthode Isomap effectue un dépliement du Swiss Roll 3D dans le plan 2D en respectant bien les distances géodésiques inter-points. On peut conclure qu'Isomap est une projection qui respecte mieux la géométrie locale des points que des méthodes linéaires comme l'ACP vue précédemment.



(a) L'espace 3D avec la distance géodésique entre les deux points.

(b) L'espace 3D avec le plus court chemin entre les deux points.

(c) L'espace de projection 2D avec les deux distances.

FIGURE 3.3 – Projection Isomap 2D d'un Swiss Roll 3D [Tenenbaum *et al.*, 2000].

3.2.4 Cartes auto adaptatives (SOM)

Elles sont aussi appelées cartes de Kohonen [Kohonen *et al.*, 2001]. Il s'agit d'une famille de techniques basées sur les réseaux de neurones artificiels avec un apprentissage non supervisé. SOM dispose de 2 espaces indépendants :

- l'espace des données (de grande dimension),
- l'espace des représentations (la carte) généralement 2D avec un topologie fixée (rectangulaire, hexagonale...). Cet espace peut être 1D (un fil), surface 3D (cylindrique, sphérique...), 3D...

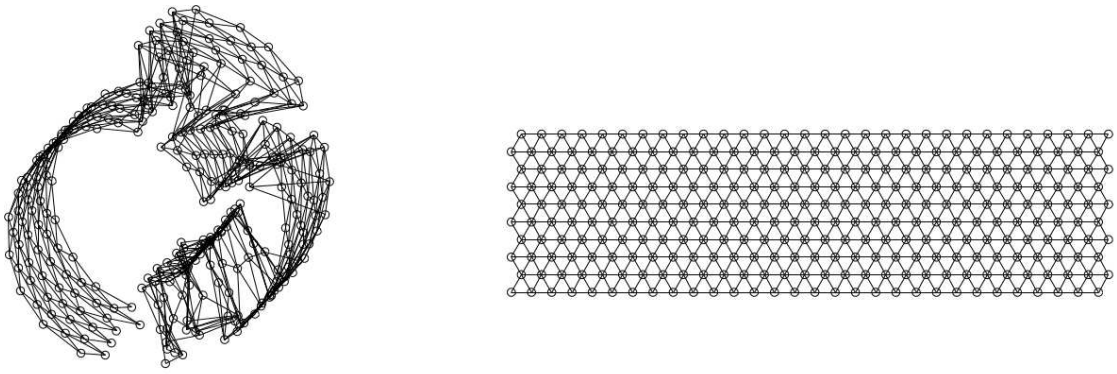


FIGURE 3.4 – Swiss Roll 3D (à gauche) déplié en 2D (à droite) par cartes auto adaptatives [Lee et Verleysen, 2002].

L'apprentissage consiste à adapter des poids (coordonnées des neurones dans l'espace des données) pour que les voisinages soient préservés au mieux tout en respectant strictement la topologie de la grille en sortie. La figure 3.4 présente la projection 2D d'un Swiss Roll 3D. Avec cet exemple, on constate que le résultat n'est pas satisfaisant en terme de respect des voisinages : de nettes ruptures apparaissent.

3.2.5 Isotop

Isotop [Lee et Verleysen, 2002] est une méthode neuronale de projection non linéaire fonctionnant avec des données de grande dimension. Elle procède en 3 étapes :

- Réduction du nombre de points par une quantification vectorielle consistant en un algorithme neuronal qui ne retient qu'un nombre restreint de centroïdes.
- Construction d'un graphe de voisinages euclidiens entre ces centroïdes.
- Projection dans l'espace de plus faible dimension en ciblant la conservation des voisinages entre les centroïdes.

Lee [Lee et Verleysen, 2002] présente une comparaison entre les techniques SOM (détaillées au paragraphe précédent) et Isotop. La figure 3.4 présente un tel exemple de projection à l'aide des SOM où le résultat est non satisfaisant. A gauche, figurent les données 3D avec une répartition « Swiss Roll » et à droite sa projection 2D en carte hexagonale. On peut voir que la projection ne respecte pas les voisinages des points sur le « Swiss Roll » : il y a des sauts entre les spires du « Swiss Roll ». Lee explique que ces résultats non satisfaisants sont la conséquence de l'apprentissage effectué dans l'espace des données combiné à la rigidité de la grille en sortie. La figure 3.5 présente le même SwissRoll projeté avec la méthode Isotop sur un plan 2D où l'on constate un bien meilleur respect des voisinages.

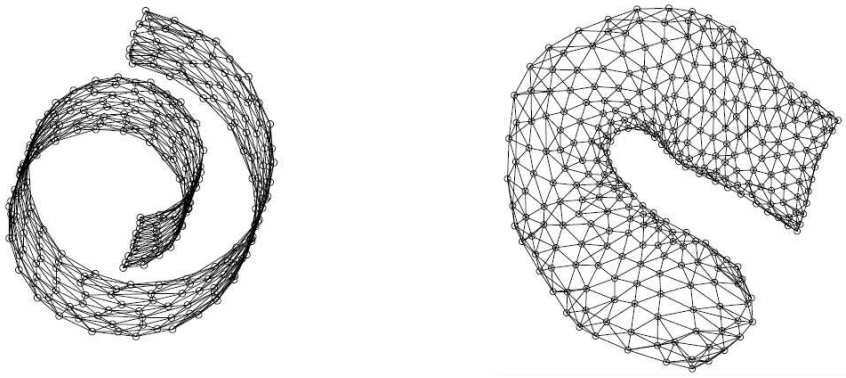


FIGURE 3.5 – Un exemple de projection Isotop [Lee et Verleysen, 2002] avec un Swiss Roll 3D (à gauche) déplié en 2D (à droite).

3.2.6 Autres techniques

Les paragraphes précédents présentent quelques techniques parmi les principales. Il en existe cependant de nombreuses autres.

Le Laplacian Eigenmap (LE) [Belkin et Niyogi, 2003] aussi nommé Spectral Embedding est une technique basée sur une décomposition spectrale de la matrice du laplacien non normalisé. Locally Linear Embedding (LLE) [Roweis et Saul, 2000] est un algorithme qui préserve la géométrie locale en minimisant une fonction de coût. LE et LLE parviennent bien à déplier des variétés définies par connexité comme le SwissRoll 3D.

Le Deep Learning [Bengio *et al.*, 2015] est un ensemble de méthodes récentes qui utilisent les réseaux de neurones artificiels. Le Deep Learning a prouvé dans les contextes de la vision par ordinateur ou de la reconnaissance de la parole qu'il pouvait donner d'excellents résultats. En particulier, les réseaux de neurones dits siamois et les réseaux de triplets

[Hoffer et Ailon, 2014] sont entraînés pour apprendre une distance et ils permettent d'effectuer des projections.

Il existe encore de nombreuses autres méthodes. Citons par exemple les méthodes à noyaux telles que Kernel PCA, Kernel Isomap [Camargo et González, 2009], le Hessian LLE [Donoho et Grimes, 2003], le Data-Driven High Dimensional Scaling [Lespinats *et al.*, 2007], le t-SNE [van der Maaten et Hinton, 2008].

3.2.7 Choix d'une projection

Selon la structure des données, nous pouvons être amenés à préférer une méthode capable de déplier des variétés définies par connexité à une méthode qui n'en est pas capable. Si l'on n'a pas cette problématique, nous pouvons préférer des méthodes purement algébriques qui sont plus rapides.

Nous pouvons mesurer la qualité de deux projections d'un même ensemble de données en évaluant des indicateurs comme les fonctions de coût évoquées dans la méthode du positionnement multidimensionnel. Nous pouvons préférer des fonctions de coût pénalisant plus le non respect de petites distances pour obtenir des projections ayant une bonne qualité au niveau local. Ou au contraire, nous pouvons choisir des fonctions de coût se focalisant plus sur le respect des grandes distances.

Toutes ces méthodes peuvent être utilisées directement pour structurer les données d'applications de visualisation 2D ou 3D. Cependant, elles peuvent aussi servir à diminuer la dimension de l'espace des données pour pouvoir ensuite exécuter une deuxième étape de structuration : la classification des données décrite au paragraphe suivant.

3.3 Structuration par classification

Lorsque le nombre d'éléments de la base est restreint, une projection 2D suivie d'un simple affichage plan est suffisant pour afficher lisiblement la base entière. Lorsque le nombre d'éléments est un peu plus conséquent, l'adjonction de mécanismes de zoom, type FishEye ou autres, permet de garder la vue d'ensemble et de se focaliser sur une partie de la base. Mais lorsque la base devient énorme ces mécanismes ne sont plus suffisants. Tous les éléments de la base ne peuvent plus être affichés ensemble de manière lisible. Il est nécessaire de résumer la base pour pouvoir l'exploiter efficacement. Conserver la vue d'ensemble, que ce soit à l'aide d'une carte physique ou mentale, est un des objectifs principaux de ces techniques.

Effectivement, dans le cas d'un nombre de données conséquent, nous avons vu dans les visualisations existantes orientées valeurs bidimensionnelles décrites dans l'annexe A des exemples d'utilisation d'une carte physique 2D qui affichent l'ensemble des informations complétées par une deuxième vue focus. Ces visualisations ne nécessitent pas de modification de la structure de la base, mais leur champ d'application est limité. Il faut que cette carte ait un sens pour l'utilisateur final. Il faut donc que les données portent intrinsèquement un repérage 2D, comme par exemple une localisation géographique. Sinon ce genre d'application nécessite un apprentissage pour que l'utilisateur intègre cette cartographie, ce que les applications expertes (médicales par exemple) peuvent exiger de leurs usagers. Ces applications qui nécessitent uniquement une structuration par projection sont donc d'usage limité. Il convient donc de s'intéresser à une deuxième façon de structurer les données : par classification.

La structuration par classification produit des classes et des partitions qui sont liées dans

un graphe. De manière très fréquente, ce graphe est un arbre qui correspond à une hiérarchie. Les Tree-Maps présentés dans l'annexe A ou les pyramides de similarités développées dans la thèse de Jau-Yuen Chen [Chen *et al.*, 2000] en sont de bons exemples. Les pyramides de similarités proposent une navigation dans des arbres quaternaires avec différents niveaux de résolution. Pour chaque niveau l, nous avons un 4^l images affichées dans une grille de taille $2^l \times 2^l$. La figure 3.6 présente ces pyramides de similarités. En (a), figure l'arbre quaternaire avec ses 3 niveaux $l = 0$, 1 et 2. En (b) et (c) figurent les visualisations correspondant aux niveaux $l = 1$ et $l = 2$.

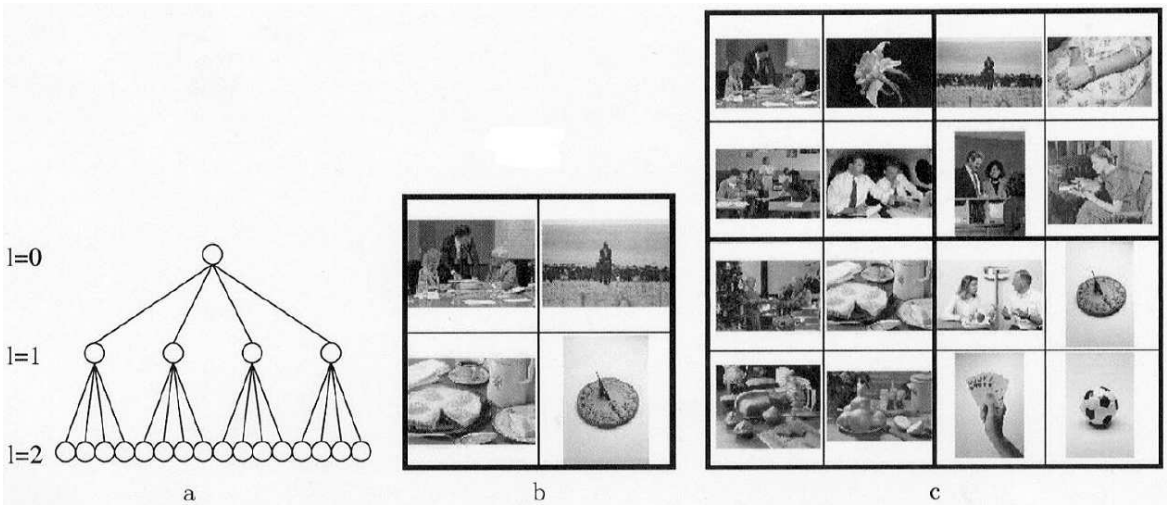


FIGURE 3.6 – Arbre quaternaire et pyramides de similarités [Chen *et al.*, 2000].

Avant d'aborder les méthodes de classification, il faut détailler le résultat qu'elles produisent : les classes, les partitions et les hiérarchies. Nous allons aussi explorer comment évaluer la qualité de ces résultats.

3.3.1 Classes, partitions et hiérarchies

3.3.1.1 Les classes

Une classe C de E est un sous-ensemble non vide de E.

Si $|E| = n$, il y a un total de $2^n - 1$ classes avec $C_n^k = \dfrac{n!}{k!(n-k)!}$ classes à $k \geq 0$ éléments.

Le livre « Éléments de classification : Aspects combinatoires et algorithmiques » est notre référence [Brucker et Barthélemy, 2007] pour toutes les notions qui suivent.

Les critères de qualité d'une classe C sont de 2 natures :

- les critères d'*homogénéité* qui mesurent la ressemblance des éléments de la classe entre eux,
- les critères de *séparabilité* qui mesurent la différence des éléments de la classe avec les éléments hors de la classe.

Si l'on considère que l'ensemble des objets E est muni d'une dissimilarité d, nous disposons des indices de qualité de la classe C suivants :

- le diamètre qui est la plus grande dissimilarité de la classe C :

$$diam_d(C) = \max\{d(x,y) \mid x,y \in C\} \tag{3.27}$$

- le rayon :

$$r_d(C) = \min\{\max\{d(x,y) \mid y \in C \setminus \{x\}\} \mid x \in C\} \tag{3.28}$$

- l'étoilement :

$$et_d(C) = \min\left\{\sum_{y\in E} d(x,y) \mid x \in C\right\} \tag{3.29}$$

- l'indice de clique :

$$cl_d(C) = \sum_{x,y\in C} d(x,y) \tag{3.30}$$

- la séparation :

$$sep_d(C) = \min\{\max\{d(x,y) \mid y \in C\} \mid x \notin C\} \tag{3.31}$$

Plus la séparation est grande, plus la classe C se distingue des autres. Tous les autres indices sont des critères d'homogénéité. Plus ils prennent une valeur petite, plus la classe est homogène.

Dans le cas particulier où $d(.)$ est la distance euclidienne, nous pouvons définir 2 autres indices d'homogénéité :

- l'inertie :

$$I(C) = \frac{1}{2|E||C|}\sum_{x,y\in C}(d(x,y))^2 \tag{3.32}$$

- la variance :

$$V(C) = \frac{1}{2|C|^2}\sum_{x,y\in C}(d(x,y))^2 \tag{3.33}$$

Plus ces 2 indices sont petits et plus la classe est resserrée autour de son *centre de gravité*.

Comparaison de classes

Si l'on doit comparer deux classes A et B identifiées par exemple avec deux algorithmes différents, l'indice de Jaccard $J(A,B) = \dfrac{|A\cap B|}{|A\cup B|}$ ainsi que les autres indices et distances décrits aux paragraphes 3.1.6 et 3.1.7 sont directement utilisables.

3.3.1.2 Les partitions

$\mathcal{P} = \{C_1, \cdots, C_p\}$ est une partition de E si :

- aucune classe n'est vide : $\forall i,\ C_i \neq \emptyset$
- la réunion de toutes les classes donne l'ensemble total : $\bigcup_{1\leq i\leq p} C_i = E$
- les classes sont disjointes : $\forall i \neq j,\ C_i \cup C_j = \emptyset$

Les indices de qualité d'une classe définis au paragraphe précédent sont étendus pour mesurer la qualité d'une partition [Brucker et Barthélemy, 2007]. Parmi les plus utilisés, nous avons :

- la somme des diamètres :

$$\sum_{1\leq i\leq p} diam_d\left(C_i\right) \tag{3.34}$$

- le diamètre maximum :

$$\max_{1\leq i\leq p} diam_d\left(C_i\right) \tag{3.35}$$

- la somme des indices de clique :

$$\sum_{1\leq i\leq p} cl_d\left(C_i\right) \tag{3.36}$$

- le maximum des indices de clique :

$$\max_{1\leq i\leq p} cl_d\left(C_i\right) \tag{3.37}$$

- la somme de l'étoilement :

$$\sum_{1\leq i\leq p} et_d\left(C_i\right) \tag{3.38}$$

- le rayon maximum :

$$\max_{1\leq i\leq p} r_d\left(C_i\right) \tag{3.39}$$

Tous ces indices sont liés à un critère d'homogénéité. Plus l'indice prend des valeurs petites, meilleure est la partition.

Dans le cas particulier où $d(.)$ est la distance euclidienne, nous avons 2 autres indices :

- l'inertie intra-classe :

$$\sum_{1\leq i\leq p} I\left(C_i\right) \tag{3.40}$$

- l'inertie inter-classe :

$$\sum_{1\leq i\leq p} \frac{|C_i|}{|E|} d\left(g\left(E\right), g\left(C_i\right)\right)^2 \quad \text{où } g(.) \text{ désigne le centre de gravité} \tag{3.41}$$

Ces critères peuvent tous les deux s'interpréter comme des critères de séparabilité. Minimiser l'inertie intra-classe revient à maximiser l'inertie inter-classe.

Comparaison de partitions

Pour évaluer les performances d'une méthode de classification, nous sommes souvent amenés à comparer la partition obtenue à celle donnant la vérité terrain. Dans ce cas, les indices décrits aux paragraphes 3.1.6 et 3.1.7 sont très souvent utilisés sous une forme adaptée pour comparer deux partitions.

Soient $\mathcal{P}$ et $\mathcal{Q}$ deux partitions de l'ensemble E avec $|E| = n$. On va examiner les paires d'éléments de E. Il y a $C_n^2 = \dfrac{n(n-1)}{2}$ paires.

Soit la fonction π qui pour une partition $\mathcal{P}$ compte le nombre de paires d'éléments classés dans la même classe C. On a alors :

$$\pi\left(\mathcal{P}\right) = \frac{1}{2} \sum_{C\in\mathcal{P}} |C|\left(|C|-1\right) \tag{3.42}$$

Définissons les deux grandeurs numériques suivantes entre les partitions $\mathcal{P}$ et $\mathcal{Q}$:

- $\pi(\mathcal{P} \wedge \mathcal{Q})$ le nombre de paires d'éléments classés dans une même classe pour $\mathcal{P}$ **et** $\mathcal{Q}$,
- $\pi(\mathcal{P} \vee \mathcal{Q})$ le nombre de paires d'éléments classés dans une même classe pour $\mathcal{P}$ **ou** $\mathcal{Q}$.

Les principaux indices utilisés pour comparer des partitions sont :

- l'indice de Jaccard :

$$\frac{\pi(\mathcal{P} \wedge \mathcal{Q})}{\pi(\mathcal{P} \vee \mathcal{Q})} \tag{3.43}$$

- l'indice de Wallace :

$$\frac{\pi(\mathcal{P} \wedge \mathcal{Q})}{\sqrt{\pi(\mathcal{P}).\pi(\mathcal{Q})}} \tag{3.44}$$

- l'indice de Johnson :

$$\frac{\pi(\mathcal{P} \wedge \mathcal{Q})}{\pi(\mathcal{P}) + \pi(\mathcal{Q})} \tag{3.45}$$

- l'indice de Rand qui mesure le pourcentage de paires en accord :

$$\frac{\pi(\mathcal{P} \vee \mathcal{Q}) - \pi(\mathcal{P} \wedge \mathcal{Q})}{n(n-1)/2} \tag{3.46}$$

- l'indice de Rand normalisé selon Hubert et Arabie [Hubert et Arabie, 1985] :

$$\frac{\pi(\mathcal{P} \wedge \mathcal{Q}) - \pi(\mathcal{P}).\pi(\mathcal{Q}) / (n(n-1)/2)}{(\pi(\mathcal{P}) + \pi(\mathcal{Q}))/2 - \pi(\mathcal{P}).\pi(\mathcal{Q}) / (n(n-1)/2)} \tag{3.47}$$

Il existe de nombreux autres indices. Ceux que nous avons présentés prennent tous leurs valeurs entre 0 et 1. La valeur 1 signifie que les deux partitions sont identiques. Par contre la valeur 0 ne correspond pas forcément à l'indépendance aléatoire des deux partitions. Pour les 4 premiers indices cette valeur est plus grande que 0. Elle dépend des effectifs des différentes classes des partitions. Seul l'indice de Rand normalisé a une valeur nulle pour deux partitions indépendantes.

3.3.1.3 Les hiérarchies

Les partitions découpent l'ensemble total en plusieurs classes disjointes et non vides. Les hiérarchies sont le regroupement de plusieurs partitions d'un même ensemble liées entre elles. Les hiérarchies sont représentées par des arbres où chaque niveau représente une partition du même ensemble.

Avant de définir précisément les hiérarchies, il faut préciser que les singletons $\{x\}$ et l'ensemble E sont appelées les classes triviales de E.

Une *hiérarchie* $\mathcal{H}$ sur un ensemble E est un ensemble de parties de E contenant toutes les classes triviales de E et telle que pour deux classes A et B quelconques de $\mathcal{H}$, on a $A \cap B \in \{\emptyset, A, B\}$.

La figure 3.7 représente une hiérarchie quelconque sur l'ensemble $E = \{a, b, c, d, e, f, g, h\}$.

La *hiérarchie triviale* représentée dans la figure 3.8 est composée uniquement des classes triviales. C'est la hiérarchie de hauteur 1.

D'une hiérarchie, on peut extraire plusieurs partitionnements : un par profondeur. Avec l'exemple de la figure 3.7, en plus du partitionnement trivial $\{\{a\}, \{b\}, \{c\}, \{d\}, \{e\}, \{f\}, \{g\}, \{h\}\}$ de la profondeur 3, on peut extraire des profondeurs 1 et 2 les partitionnements $\{\{a, b\}, \{c\}, \{d, e, f, g, h\}\}$ et $\{\{a, b\}, \{c\}, \{d, e\}, \{f, g, h\}\}$.

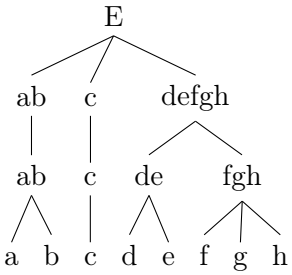


FIGURE 3.7 – Une hiérarchie quelconque.

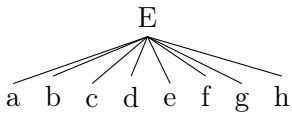


FIGURE 3.8 – La hiérarchie triviale.

Et réciproquement, les *partitionnements* peuvent être considérés comme des hiérarchies. Ce sont les hiérarchies de hauteur 2. La figure 3.9 est un exemple de partitionnement.

Les *chaînages* sont les hiérarchies dont toutes les paires de classes non triviales ont une intersection non vide. La figure 3.10 est un exemple de chaînage.

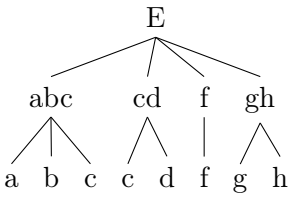


FIGURE 3.9 – Un partitionnement.

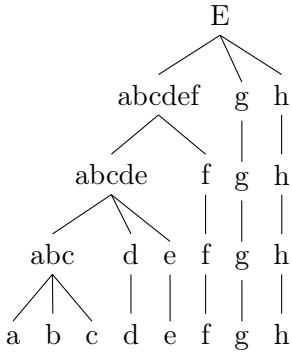


FIGURE 3.10 – Un chaînage.

Les *hiérarchies binaires* sont les hiérarchies dont toutes les classes non singletons ont deux fils. Les *hiérarchies binaires équilibrées* sont les hiérarchies binaires dont le nombre d'éléments de toute classe est la moitié du nombre d'éléments de la classe père. Les *peignes* sont les chaînages binaires. Les figures 3.11 et 3.12 donnent des exemples de telles hiérarchies.

En analyse de données, les méthodes de classifications sont intéressantes car elles permettent de regrouper les données ressemblantes par paquets. Les résultats de ces classifications sont des partitionnements. Il faut pouvoir discuter la qualité de ces résultats. Nous avons vu ce que sont les partitions et comment évaluer leur qualité ou les comparer à d'autres partitions qui peuvent être des vérités terrain. Ensuite nous avons vu les hiérarchies. Elles fournissent une structure intéressante qui permet d'avoir plusieurs partitions hiérarchisées. À partir d'une hiérarchie, nous pouvons choisir le niveau de cette hiérarchie qui fournit la partition la plus adaptée à ce que l'utilisateur veut. Les hiérarchies sont aussi des structures intéressantes car elles sont la base des méthodes de classification hiérarchique. Nous nous intéressons maintenant à ces méthodes de classification qui peuvent être de différentes natures. Elles peuvent être automatiques ou au contraire utiliser une connaissance humaine. C'est ce que nous examinons dans la suite.

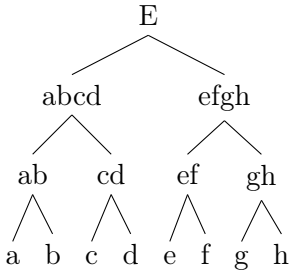


FIGURE 3.11 – Une hiérarchie binaire équilibrée.

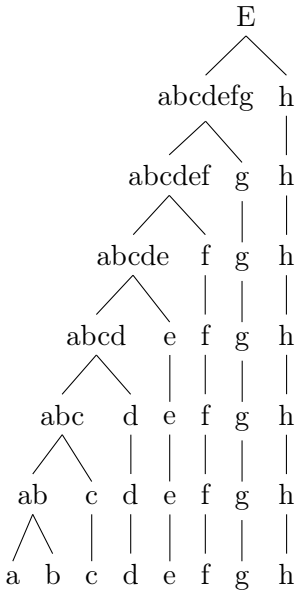


FIGURE 3.12 – Un peigne.

3.3.2 Classification automatique

La classification automatique ou classification non supervisée est appelée clustering en anglais. Elle consiste à attribuer une classe à chaque objet à classer. Elle est non supervisée et totalement automatique en ne nécessitant pas de connaissance. Ceci la rend intéressante car elle ne nécessite pas d'intervention humaine.

Pour obtenir une hiérarchie, la méthode la plus utilisée est la Classification Ascendante Hiérarchique (CAH) que nous décrivons au paragraphe 3.3.2.1. D'autres techniques moins classiques existent. Citons l'algorithme des voisins réciproques [Brucker et Barthélemy, 2007] qui peut être vu comme une amélioration de la CAH. L'amélioration permet de passer de la complexité algorithmique en $O(n^3)$ de la CAH à un $O(n^2)$.

Pour obtenir un partitionnement, les « k-means » sont une méthode classique que nous présentons au paragraphe 3.3.2.2. Beaucoup d'autres techniques existent et les techniques de projections du paragraphe 3.2 peuvent aussi être adaptées pour construire un partitionnement. Par exemple, le Clustering Spectral que nous décrirons précisément au paragraphe 3.4 est une méthode qui commence par une projection des données dans un espace de plus faible dimension.

Citons aussi l'algorithme des transferts [Brucker et Barthélemy, 2007] qui, en partant d'une partition initiale (choisie aléatoirement par exemple), effectue des transferts d'objets entre les classes pour obtenir un partitionnement optimal. Ces transferts sont guidés par l'optimisation d'une fonction objectif. Cette fonction peut être l'un des indices du paragraphe 3.3.1.2 comme la somme de l'indice de clique.

La plupart des techniques de classification nécessitent de fixer le nombre de classes a priori. Certaines méthodes peuvent estimer ou déduire le nombre de classes comme par exemple dans les techniques d'Azran [Azran, 2006] qui estiment le nombre de classes à partir des valeurs propres de la matrice de distance.

Nous ne présentons ici que les principales méthodes ainsi que celles que nous utilisons

dans cette thèse. Il existe de nombreuses autres méthodes bien décrites dans des ouvrages comme [Maimon et Rokach, 2005].

3.3.2.1 La Classification Ascendante Hiérarchique

L'algorithme de Classification Ascendante Hiérarchique (CAH) décrit dans de nombreux ouvrage comme [Brucker et Barthélemy, 2007] suppose que l'on a une dissimilarité d entre les objets à classifier. La méthode est hiérarchique car elle produit une hiérarchie. L'algorithme est ascendant car il construit la hiérarchie en partant de la plus grande profondeur pour remonter à la racine de la hiérarchie de la façon suivante :

1. la partition triviale de l'ensemble $E = \{x_1, \cdots, x_n\}$ est construite,
2. les 2 classes les plus proches sont fusionnées pour obtenir une nouvelle partition,
3. l'étape 2 est répétée $n-1$ fois car au bout de $n-1$ fusions, nous obtenons l'ensemble E et il n'y a plus rien à fusionner.

Lors de l'étape 2, les 2 classes les plus proches sont celles qui ont la dissimilarité nommée indice d'agrégation, la plus faible.

Cette dissimilarité inter-classes est définie à l'aide de la dissimilarité entre les objets. Les dissimilarités entre les singletons sont définies de la façon suivante :

$$d(\{x\}, \{y\}) = d(x, y) \tag{3.48}$$

Ensuite lors des opérations de fusion, la dissimilarité fusionnée est définie selon l'une des 3 techniques suivantes :

$$\text{le lien simple} \quad : \quad d(A, B) = \min_{x \in A, y \in B} (d(x, y)) \tag{3.49}$$

$$\text{le lien moyen} \quad : \quad d(A, B) = \frac{1}{|A||B|} \sum_{x \in A} \sum_{y \in B} (d(x, y)) \tag{3.50}$$

$$\text{le lien complet} \quad : \quad d(A, B) = \max_{x \in A, y \in B} (d(x, y)) \tag{3.51}$$

Le résultat de la classification est une hiérarchie avec des indices d'agrégation. Ceci se représente sous la forme d'un dendrogramme comme celui de la figure 3.13.

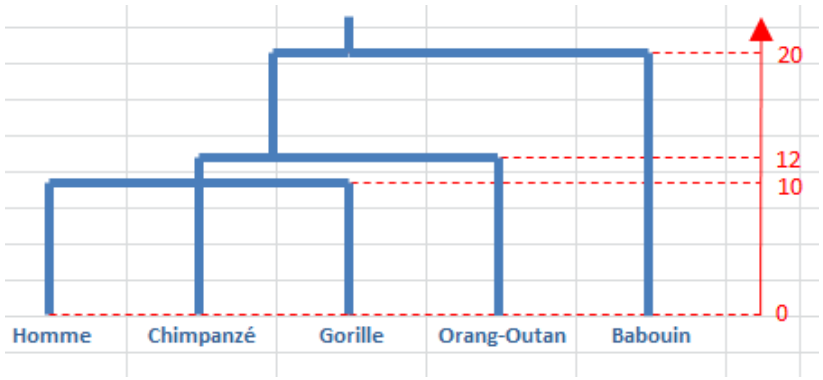


FIGURE 3.13 – Exemple de dendrogramme avec les principaux hominidés obtenu avec un programme original en VBA sous Microsoft Office Excel ©.

3.3.2.2 Les k-means

L'algorithme des k-means [Brucker et Barthélemy, 2007] est aussi appelé algorithme des centres mobiles. La méthode suppose que nous ayons n points $x_1, \cdots, x_n$ d'un ensemble $\mathbb{R}^N$ muni de la distance euclidienne.

Soit k le nombre de classes voulu. Les étapes de l'algorithme des k-means sont :

1. k points $c_1, \cdots, c_k$ sont choisis (aléatoirement par exemple) comme étant les centres des k classes $C_1, \cdots, C_k$,
2. chaque nouveau point x_i est rattaché à la classe C_j dont le centre c_j est le plus proche,
3. les centres $c_1, \cdots, c_k$ sont mis à jour comme étant les barycentres des classes $C_1, \cdots, C_k$,
4. les étapes 2 et 3 sont répétées jusqu'à ce que les classes ne changent plus.

Il existe plusieurs variantes de l'algorithme [Brucker et Barthélemy, 2007] comme par exemple le online k-means qui recalcule les centres de gravité de l'étape 3 à chaque fois qu'un point est examiné lors de l'étape 2.

La classification automatique peut dans des situations difficiles donner des résultats insuffisants. Pour résoudre ce problème, l'ajout de connaissance se présente comme une solution adaptée que nous examinons dans les paragraphes suivants.

3.3.3 Classification supervisée

En anglais, la classification supervisée est simplement nommée « classification » par opposition au « clustering » qui désigne la classification non supervisée ou automatique.

La classification supervisée est caractérisée par l'existence d'un *ensemble d'apprentissage* sur lequel le classement est totalement connu. Cette vérité terrain est utilisée pour effectuer l'apprentissage de règles de classement. Ensuite, lorsque l'on veut évaluer la qualité de la méthode de classification, on utilise un deuxième échantillon que l'on appelle *ensemble de test*. Il existe de nombreuses méthodes décrites dans des ouvrages comme [Hastie *et al.*, 2009]. Nous décrivons dans la suite les méthodes qui nous servent de références dans nos expérimentations des chapitres 4 et 5.

3.3.3.1 Méthodes des k plus proches voisins

La méthode des k-NN est décrite dans de nombreux ouvrages comme par exemple dans le livre [Hastie *et al.*, 2009]. C'est sans doute la technique la plus simple : chaque donnée de l'ensemble de test est classée dans la classe majoritaire de ses k plus proches voisins appartenant à l'ensemble d'apprentissage.

Pour appliquer cette méthode, il faut disposer d'une métrique et il faut fixer ou déterminer le paramètre k.

La complexité de l'algorithme exhaustif est en $O(dn)$ où d est la dimension de l'espace et n le nombre d'objets. Cet algorithme est facilement parallélisable. Cependant différentes stratégies permettent d'optimiser cette méthode dans différents cas particuliers. Par exemple, lorsque la dimension d est petite, il est possible d'effectuer des prétraitements pour structurer les données sous forme d'arbre et obtenir ainsi des algorithmes basés sur des parcours d'arbres qui sont plus efficaces. Avec l'utilisation des arbres kd [Bentley, 1975] la complexité de recherche des k plus proches voisins est en $O(d \log(n))$.

3.3.3.2 Machine à vecteurs de support

La méthode Support Vector Machine (SVM) [Hastie *et al.*, 2009] consiste à déterminer un hyperplan qui sépare les classes et qui est le plus éloigné possible des objets. Ceci consiste à maximiser les marges. La figure 3.14 présente différentes coupes séparatrices possibles entre les deux classes. La figure 3.15 présente la coupe optimale et sa marge maximale.

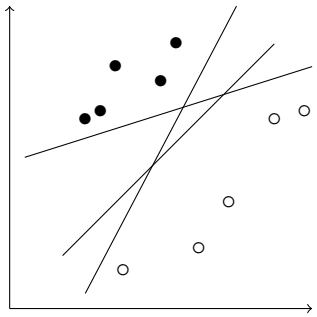


FIGURE 3.14 – Différentes coupes séparatrices.

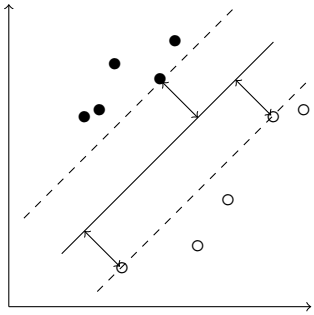


FIGURE 3.15 – La coupe optimale en trait plein avec sa marge maximale.

Lorsque la recherche de l'hyperplan optimal est effectuée dans l'espace des données E comme dans le cas précédent, on parle de SVM linéaire (équation 3.52). Cependant, les données peuvent ne pas être séparables selon un hyperplan dans E. Dans ce cas, il existe d'autres formes de SVM utilisables. Si l'on considère une transformation non linéaire ϕ, l'espace de départ E est transformé en un nouvel espace $\phi(E)$ appelé espace de redescription. Des données non séparables dans E peuvent devenir séparables dans un espace $\phi(E)$ de plus grande dimension.

La recherche de marge maximale dans $\phi(E)$ est équivalente à la recherche de marge maximale dans E en substituant au produit scalaire de l'équation 3.52, la fonction noyau $K(x, y) = \phi(x)^T.\phi(y)$. Dans la pratique, c'est cette optimisation à l'aide de la fonction noyau K qui est effectuée car une optimisation dans l'espace $\phi(E)$ de grande dimension serait plus coûteuse.

Selon le noyau utilisé, on parle de différentes méthodes SVM dont voici quelques exemples :

- le *SVM linéaire* correspond à l'utilisation du noyau linéaire :

$$K(x, y) = x^T.y \tag{3.52}$$

- le *SVM RBF* correspond à l'utilisation du noyau gaussien :

$$K(x, y) = \exp\left(-\gamma\|x - y\|^2\right) \tag{3.53}$$

- le *SVM Khi2* correspond à l'utilisation du noyau Khi2 :

$$K(x, y) = \exp\left(-\gamma \sum_i \frac{(x_i - y_i)^2}{x_i + y_i}\right) \tag{3.54}$$

3.3.3.3 Forêts aléatoires

La méthode des *Forêts aléatoires* [Breiman, 2001] aussi appelées *Forêts d'arbres décisionnels* est basée sur les arbres de décisions binaires. Une forêt aléatoire est composée d'un ensemble d'arbres dans lesquels est introduit de l'aléatoire.

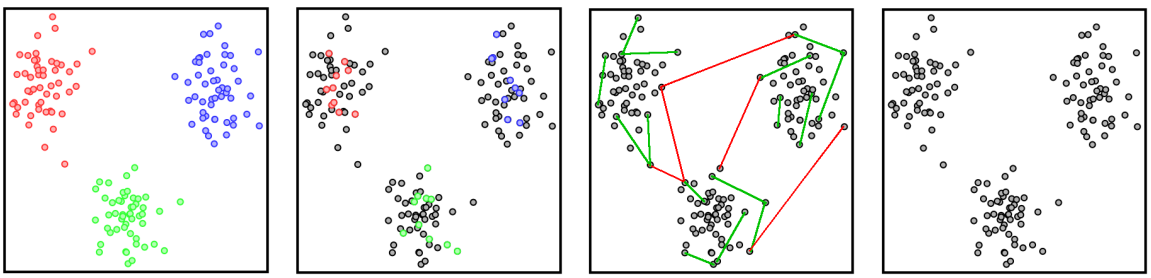
De nombreux modèles de forêts aléatoires existent. Ils diffèrent sur la manière d'incorporer l'aléatoire. Par exemple, Breiman a mis au point une méthode de forêts aléatoires en 2001 [Breiman, 2001]. Cette méthode est basée sur l'algorithme CART qui procède en deux phases :

- la phase d'expansion construit un arbre de décision maximal en considérant une multitude de sous échantillons de l'ensemble d'apprentissage. Sur chacun de ces sous échantillons, un arbre de décision binaire est construit en ne retenant qu'une partie des variables tirées aléatoirement. L'arbre maximal est construit en retenant les décisions majoritaires des différents arbres.
- la phase d'élagage réduit l'arbre tout en conservant des résultats proches de l'arbre maximal. Cette réduction vise à obtenir un arbre plus léger et efficace mais suffisamment fiable. La réduction de l'arbre est effectuée en remplaçant des branches très arborescentes et feuillues qui conduisent à une décision presque certaine par une seule feuille portant cette décision.

3.3.4 Classification semi-supervisée

Dans des situations difficiles, la classification automatique peut donner des résultats insuffisants voir même inappropriés. La classification supervisée permet de résoudre ce problème en utilisant un étiquetage complet sur un échantillon des données. Cet étiquetage nécessite une intervention humaine préalable à la classification qui peut s'avérer fastidieuse et coûteuse. Il est donc intéressant d'examiner une démarche intermédiaire qui n'utilise qu'une connaissance partielle : la classification semi-supervisée.

Jain [Jain, 2010] décrit les deux principaux types de connaissance partielle : la connaissance partiellement étiquetée et la connaissance partiellement contrainte par paires d'objets. Comme on peut le voir sur l'exemple de la figure 3.16, la connaissance partiellement étiquetée est simplement un étiquetage sur uniquement une partie des objets, c'est-à-dire l'appartenance par classe pour seulement une partie des données. La connaissance partiellement contrainte par paires d'objets est la donnée d'un certain nombre de contraintes « Must Link » en vert et « Cannot Link » en rouge qui indiquent simplement si les deux objets sont de la même classe ou non. Ces deux types de connaissance sont les principaux, mais ce ne sont pas les seuls. Par exemple Law, Thome et Cord [Law *et al.*, 2013] utilisent une connaissance par quadruplets d'objets qui indique si les 2 premiers objets sont plus similaires entre eux que les 2 derniers objets. Cette connaissance générale peut être aussi restreinte à des triplets.



(a) Totalement étiquetée. (b) Partiellement étiquetée. (c) Partiellement contrainte. (d) Aucune connaissance.

FIGURE 3.16 – Les différents types de connaissance selon Jain [Jain, 2010]

Le tableau 3.2 recense les différents types de classification en fonction des différents types

de connaissance utilisée, avec les correspondances français/anglais. Si l'on n'utilise aucune connaissance, nous avons à faire à une classification automatique ou clustering. Si l'on effectue un apprentissage sur un ensemble de développement où l'on dispose d'un étiquetage total, pour ensuite effectuer un classement sur un ensemble de test, il s'agit d'une classification supervisée. Cependant certaines méthodes peuvent aussi apprendre de données non étiquetées en utilisant par exemple les positions des données étiquetées et non étiquetées. Si une méthode adopte ce genre de démarche en utilisant un ensemble d'apprentissage partiellement étiqueté, nous avons à faire à une méthode de classification semi-supervisée. Et pour finir, si une méthode de clustering est adaptée de manière à prendre en compte une connaissance partiellement étiquetée ou partiellement contrainte, sans apprentissage, il s'agira d'un clustering semi-supervisé. Comme on peut le voir dans le tableau 3.2, il existe des ambiguïtés dans l'utilisation des termes classification et semi-supervisé. C'est pourquoi il semble préférable de privilégier les termes en gras du tableau dans leur contexte d'utilisation.

<table>
<tr><th rowspan="2">Type de connaissance</th><th colspan="2">Type de classification</th></tr>
<tr><th>en anglais</th><th>en français</th></tr>
<tr><td>Totalement étiquetée</td><td>Classification</td><td>Classification supervisée</td></tr>
<tr><td rowspan="2">Partiellement étiquetée</td><td>Semi-supervised classification</td><td rowspan="2">Classification semi-supervisée</td></tr>
<tr><td rowspan="2">Partiellement contrainte</td><td>Semi-supervised clustering</td></tr>
<tr><td>Aucune</td><td>Clustering</td><td>Classification automatique</td></tr>
</table>

TABLE 3.2 – Les types de classification en anglais et en français en fonction du type de connaissance utilisée.

Il existe de nombreuses méthodes de classification semi-supervisée. Il s'agit d'adaptations des méthodes de classification supervisée pour prendre en compte la connaissance donnée sous forme d'un étiquetage partiel tout en tirant profit des données non étiquetées. Le premier exemple est le co-apprentissage [Blum et Mitchell, 1998] qui consiste à utiliser deux classifieurs qui sont entraînés sur deux ensembles de caractéristiques différents et idéalement indépendants. Pour chaque classifieur, les données non étiquetées dont l'étiquette prédite semble la plus sûre sont données en apprentissage à l'autre. Et lorsque l'apprentissage est terminé, les deux classifieurs sont combinés. Le principe du co-apprentissage est que deux classifieurs entraînés selon deux projections indépendantes d'un même espace de données doivent étiqueter de manière identique. Il existe aussi la méthode de SVM transductif [Joachims, 1999] qui est une adaptation de la méthode SVM de façon à utiliser aussi bien les données étiquetées que les données non étiquetées pour maximiser la marge séparatrice.

Il existe aussi de multiples méthodes de clustering semi-supervisé. Il s'agit d'adaptations des méthodes de clustering pour prendre en compte la connaissance donnée sous forme d'étiquetages ou de contraintes :

- Avec des étiquetages absolus, il existe des adaptations des k-means capables de les prendre en compte lors de l'initialisation des centroïdes [Basu *et al.*, 2002]. Dans l'algorithme Seeded-KMEANS, les objets étiquetés sont uniquement utilisés pour déterminer les centroïdes initiaux. L'algorithme est ensuite exactement celui des k-means originaux. Dans l'algorithme Constrained-KMEANS, les objets étiquetés sont utilisés pour déterminer les centroïdes initiaux, puis affectés de manière définitive à un clus-

ter. Ensuite, seuls les objets non étiquetés peuvent changer de cluster. Le choix entre les deux algorithmes dépend de la connaissance que nous avons sur le bruit dans les données. Si les données ne sont pas bruitées, le constrained-KMEANS peut être utilisé. Dans le cas contraire, il faut privilégier le Seeded-KMEANS pour autoriser les changements de clusters à tous les objets. Il existe d'autres méthodes où la connaissance est intégrée dans les étapes suivantes de l'algorithme des k-means. Par exemple, Demiriz [Demiriz *et al.*, 1999] utilise un algorithme génétique pour calculer les centroïdes. Cet algorithme cherche à optimiser une fonction dépendant de la qualité des classes calculée sur les objets dont on connaît l'étiquetage.

- Plutôt qu'obtenir des annotations absolues par un expert, on peut lui demander des annotations de type « ressemblance » (« Must Link ») et différence (« Cannot Link »). Aussi une connaissance donnée par un étiquetage partiel peut être transformée en une connaissance par paires d'objets de type « Must Link » et « Cannot Link ». Lorsque nous avons un étiquetage partiel, il est donc possible d'utiliser aussi toutes les méthodes intégrant une connaissance par paires. Ces méthodes sont nombreuses. Citons le Constrained K-means Clustering (COP-KMEANS) [Wagstaff *et al.*, 2001] et le MPCK-MEANS [Bilenko *et al.*, 2004] qui modifient l'algorithme des k-means pour lui permettre d'intégrer de la supervision à l'aide de contraintes ML et CL. COP-KMEANS modifie l'algorithme des k-means de manière à respecter à chaque itération les contraintes entre paires d'objets données. Davidson et Ravi [Davidson et Ravi, 2005] ont proposé un algorithme de clustering hiérarchique agglomératif contraint qui permet lui aussi d'intégrer des contraintes par paires d'objets.

3.3.5 Classification semi-supervisée interactive et active

Les méthodes semi-supervisées, en s'inscrivant dans un processus itératif, peuvent devenir interactives. Comme représentée dans la figure 3.17, une classification peut être complétée par l'appel à un expert, nommé parfois Oracle, qui ajoute de la connaissance au fur et à mesure des itérations. Ce processus peut être intégré dans une application de classification semi-supervisée dont le seul but est de classer une base à l'aide d'un opérateur humain. Dans les méthodes de clustering semi-supervisé interactif, le schéma le plus couramment rencontré est de soumettre des paires d'objets à l'expert afin qu'il puisse statuer s'il s'agit de contraintes « Must Link » ou « Cannot Link ».

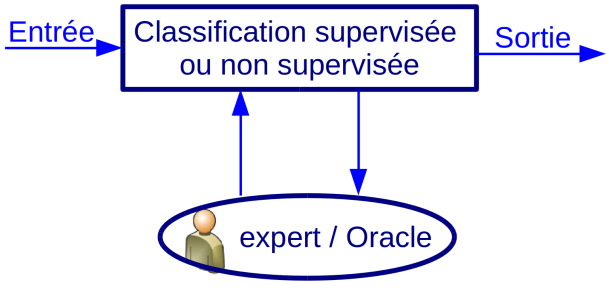


FIGURE 3.17 – Un schéma de classification semi-supervisée interactive.

Nous venons d'utiliser les termes expert et Oracle pour dénommer l'entité qui va délivrer la connaissance. Dans la littérature, nous rencontrons couramment le terme d'expert. Un expert est une « personne apte à juger quelque chose, un connaisseur » (Dictionnaire Larousse). C'est

bien d'un expert du domaine dont nous avons besoin pour recueillir la connaissance. Dans le cas, d'une approche collaborative, il est possible que des experts rentrent en conflit dans la connaissance qu'ils dispensent. Et un unique expert peut entrer en conflit avec lui-même. Dans le cas d'une connaissance dispensée sous forme d'étiquetage, les conflits peuvent être évités simplement en ne demandant pas d'étiqueter deux fois le même objet. Par contre dans le cas d'une connaissance par paire, les conflits peuvent apparaître facilement. Par exemple, si on a trois objets et que l'on demande de superviser les trois paires possibles et que l'on obtient deux Must-Link et un Cannot-Link, on obtient un conflit. Effectivement les deux Must-Link nous indiquent que les trois objets sont de la même classe. Ce qui est contredit par la troisième contrainte Cannot-Link. Des configurations conflictuelles plus complexes existent aussi.

Le terme Oracle désigne une « décision jugée infaillible et émanant d'une personne de grande autorité » ou « personne considérée comme infaillible » (Dictionnaire Larousse). Donc l'emploi d'un Oracle nous prémunirait de tout conflit de connaissance. Dans la pratique, pour que cet Oracle considéré infaillible le demeure, il ne faut pas le mettre dans une situation où il pourrait faillir. Reste à savoir comment effectivement être prémuni de tout conflit. Dans le cas, d'un étiquetage par classe, il suffit de considérer la réponse de l'Oracle comme infaillible et de ne jamais demander d'étiqueter deux fois le même objet. Dans le cas de contraintes par paires la tâche semble plus compliquée. La solution existe pourtant. Avant de soumettre une paire à l'Oracle, il suffit d'examiner si cette supervision peut créer une contrainte incohérente. Et si c'est le cas, il faut s'abstenir de la soumettre à l'Oracle.

En envisageant quelles paires d'objets soumettre ou non à l'Oracle, nous venons d'ajouter une étape dans le processus interactif entre la classification et la supervision : la sélection des paires. Nous obtenons un schéma d'apprentissage actif des contraintes, représenté par la figure 3.18. Cependant, garantir que l'Oracle demeure infaillible n'est qu'un des cas d'usage du schéma de sélection active des contraintes. Il existe de nombreuses stratégies de sélection des contraintes qui visent à diminuer la sollicitation de l'Oracle et augmenter la vitesse de convergence de la méthode itérative.

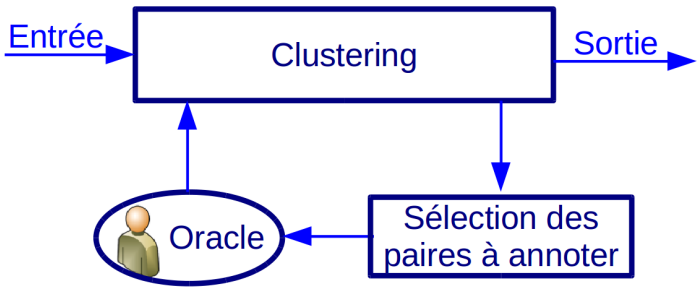


FIGURE 3.18 – Clustering semi-supervisé actif.

Pour comparer plusieurs méthodes de clustering semi-supervisé, on les inscrit dans un schéma actif où les paires à annoter sont souvent sélectionnées aléatoirement parmi toutes les paires d'objets possibles. Cependant, Davidson [Davidson *et al.*, 2006] a démontré que dans certains cas, des paires mal choisies donnent des contraintes qui dégradent la qualité du clustering et qu'à l'inverse, des paires bien choisies peuvent donner des contraintes qui améliorent le résultat de manière significative. Davidson introduit deux mesures pour quantifier l'utilité des contraintes : l'*informativité* qui mesure la quantité d'information apportée et la *cohérence* qui mesure l'accord entre les différentes contraintes.

Vu, Labroche et Bouchon-Meunier [Vu *et al.*, 2012] proposent un modèle de sélection

active des contraintes qui permet d'identifier et classer des arêtes critiques dans le graphe des plus proches voisins. Ce sont des arêtes « passerelles » entre des groupes de points fortement connectés qui sont intéressantes à soumettre à l'avis d'un expert pour savoir si l'on doit couper ou non le graphe à cet endroit. Une approche similaire [Xiong *et al.*, 2014] propose d'identifier les objets les plus ambigus et de sélectionner les liens issus de ces objets. Une des originalités de ce travail réside dans le fait que cette sélection d'objet cherche à se focaliser sur les liens qui ont les plus grandes chances d'apporter des changements significatifs dans les résultats du clustering.

Ces modèles utilisent en général une sélection a posteriori des contraintes. À chaque itération, les paires d'objets sont sélectionnées en fonction du clustering obtenu à l'itération précédente. Les algorithmes ont donc un coût de calcul élevé, mais ils s'avèrent efficaces d'un point de vue de la qualité de la partition obtenue.

3.3.6 Synthèse

Dans cette section, nous avons présenté la classification supervisée, la classification automatique (clustering) ainsi que la classification semi-supervisée et le clustering semi-supervisé. Nous avons vu qu'en incluant le clustering semi-supervisé dans un processus itératif avec une étape de supervision, il devient interactif. Dans le cas général du clustering semi-supervisé interactif, le superviseur peut être amené à examiner le résultat du clustering et décider quoi et comment superviser. Le clustering semi-supervisé actif est un cas particulier où une étape de sélection automatique de la connaissance à présenter au superviseur est ajoutée : le superviseur ne choisit pas ce qu'il doit superviser. Pour la suite, nous nous intéressons à une méthode de clustering particulière : le Clustering Spectral.

3.4 Le Clustering Spectral

Les performances d'un classifieur sont fortement dépendantes des propriétés et de la structure des données. Lorsque l'on a des clusters convexes, les méthodes classiques telles que les k-means donnent de bons résultats. Cependant, comme illustrées dans la partie gauche de la figure 3.19, ces méthodes ne sont pas capables d'identifier des variétés caractérisées par une connectivité complexe des données. Dans de telles situations, d'autres algorithmes sont opérationnels comme Isomap, le positionnement multidimensionnel (MDS) et le Clustering Spectral (voir la partie droite de la figure 3.19). Ces méthodes tentent généralement d'identifier un espace de dimension inférieure qui représente et sépare bien les données. Nous explorons dans cette section le Clustering Spectral [von Luxburg, 2007] qui est capable de fonctionner efficacement sans hypothèse de forme sur les clusters. De plus, cette méthode peut traiter de grands volumes de données en s'appuyant sur des graphes de similarité sparses (des matrices creuses). Ce cas de figure est une caractéristique intéressante pour nos travaux portant sur de grandes bases de données vidéos.

Dans sa forme classique que nous présentons au paragraphe 3.4.2, le Clustering Spectral est une méthode totalement automatique qui n'utilise que les données fournies en entrée. Or dans des situations complexes, comme l'analyse ou la compréhension de vidéos, le fossé sémantique entre les caractéristiques de bas niveau extraites et la classification haut niveau attendue est très grand. L'introduction de connaissance pour guider le Clustering Spectral présente alors son intérêt. En suivant cette idée, nous pouvons définir un Clustering Spectral semi-supervisé par l'ajout d'un faible nombre de contraintes par paires présenté au paragraphe

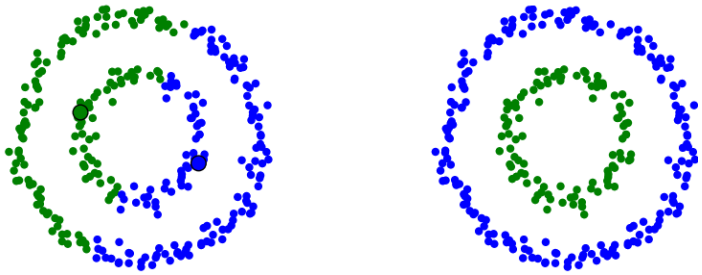


FIGURE 3.19 – Les 2 classes obtenues en utilisant la méthode des k-means à gauche et le Clustering Spectral à droite.

3.3.4. De plus, en choisissant bien certaines contraintes dans l'ensemble des données, on peut obtenir de bons résultats de clustering [Davidson *et al.*, 2006] en se focalisant sur les ambiguïtés et en diminuant les coûts d'annotation.

Voici tout d'abord au paragraphe 3.4.1 quelques bases de la théorie spectrale des graphes illustrées par des exemples.

3.4.1 Théorie spectrale des graphes

Luxburg présente dans son tutoriel [von Luxburg, 2007] la théorie spectrale des graphes utilisée par le Clustering Spectral.

3.4.1.1 Graphes pondérés non orientés

Soit le graphe non orienté $G = (V, E)$. G est composé de n sommets ou nœuds v_i (vertices en anglais), et de m arêtes e_k (edges en anglais). Pour chaque arête de sommets i et j, la pondération est $w_{ij} = w_{ji} \geqslant 0$; il s'agit de similarités. $w_{ij} = 0$ signifie qu'il n'y a pas d'arête entre les sommets i et j. Plus w_{ij} est grand, plus les sommets i et j sont similaires. La figure 3.20 présente un tel exemple de graphe.

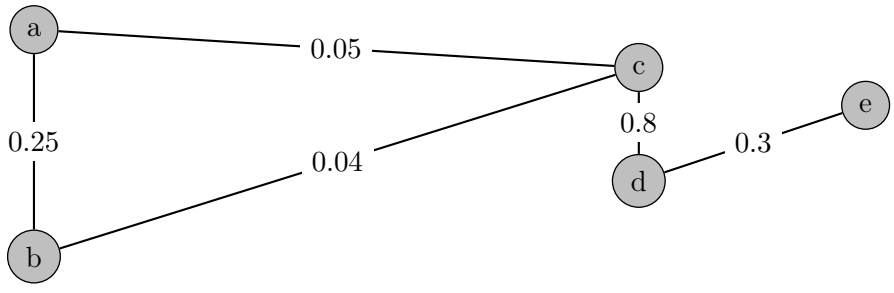


FIGURE 3.20 – Un exemple de graphe pondéré à 5 nœuds.

3.4.1.2 Matrice d'adjacence et degrés

La matrice d'adjacence qui représente l'ensemble des similarités entre chaque nœud est

$$W = (w_{ij})_{1 \leq i,j \leq n} \tag{3.55}$$

La matrice d'adjacence du graphe 3.20 est

$$W = \begin{pmatrix} 1 & 0.25 & 0.05 & 0 & 0 \\ 0.25 & 1 & 0.04 & 0 & 0 \\ 0.05 & 0.04 & 1 & 0.8 & 0 \\ 0 & 0 & 0.8 & 1 & 0.3 \\ 0 & 0 & 0 & 0.3 & 1 \end{pmatrix} \tag{3.56}$$

La matrice des degrés est $D = diag(d_1, \ldots, d_n)$ où les degrés d_i correspondent à la somme des poids des liens du nœud i avec

$$d_i = \sum_{j=1}^{n} w_{ij} > 0 \tag{3.57}$$

La matrice des degrés du graphe 3.20 est

$$D = \begin{pmatrix} 1.3 & 0 & 0 & 0 & 0 \\ 0 & 1.29 & 0 & 0 & 0 \\ 0 & 0 & 1.89 & 0 & 0 \\ 0 & 0 & 0 & 2.1 & 0 \\ 0 & 0 & 0 & 0 & 1.3 \end{pmatrix} \tag{3.58}$$

3.4.1.3 Sous-graphes et coupe de graphes

Il est intéressant d'isoler en sous-graphes les nœuds similaires en vue de les classer. Un sous-graphe $A \subset V$ est caractérisé par le vecteur

$$\mathbf{1}_A = (f_1, \ldots, f_n)^T \in \mathbb{R}^n \quad avec \quad f_i = \begin{cases} 1 & si \quad v_i \in A \\ 0 & sinon \end{cases} \tag{3.59}$$

Mais pour isoler ces sous-graphes, les coupes ont un coût. Le coût d'une coupe entre deux sous-graphes A et B de V est

$$cut(A, B) = \sum_{i \in A, j \in B} w_{ij} \tag{3.60}$$

La figure 3.21 présente un exemple de coupe du graphe 3.20 en deux sous-graphes $A = \{a; b\}$ et $B = \{c; d; e\}$. Sur cet exemple, nous avons $\mathbf{1}_A = (1, 1, 0, 0, 0)^T$, $\mathbf{1}_B = (0, 0, 1, 1, 1)^T$ et $cut(A, B) = 0.04 + 0.05 = 0.09$.

3.4.1.4 Matrice laplacienne

La matrice laplacienne non-normalisée est

$$L = D - W \tag{3.61}$$

Cette matrice laplacienne a de nombreuses propriétés et applications [Mohar, 1997]. Historiquement, l'une de ses premières utilisations est le théorème de Kirchhoff qui permet de déterminer le nombre d'arbres couvrants d'un graphe [Mohar, 1997]. Dans le cas du Clustering Spectral, elle permet de déterminer les composantes connexes du graphe et de décider où couper le graphe.

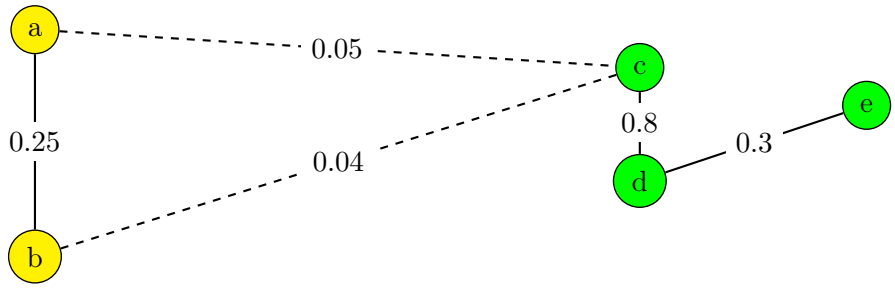


FIGURE 3.21 – Exemple de coupe du graphe 3.20 en deux sous-graphes $A = \{a; b\}$ et $B = \{c; d; e\}$.

La matrice laplacienne du graphe 3.20 est

$$L = \begin{pmatrix} 0.3 & -0.25 & -0.05 & 0 & 0 \\ -0.25 & 0.29 & -0.04 & 0 & 0 \\ -0.05 & -0.04 & 0.89 & -0.8 & 0 \\ 0 & 0 & -0.8 & 1.1 & -0.3 \\ 0 & 0 & 0 & -0.3 & 0.3 \end{pmatrix} \tag{3.62}$$

La matrice laplacienne L est symétrique et sa première propriété [von Luxburg, 2007] est que pour tout vecteur $f \in \mathbb{R}^n$ on a :

$$f^T L f = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \left(f_i - f_j\right)^2 \tag{3.63}$$

Cette propriété 3.63 est importante car toutes les propriétés qui suivent en découlent.

3.4.1.5 Le spectre de la matrice laplacienne

Les valeurs propres λ_i et les vecteurs propres associés u_i de L vérifient

$$\lambda_1 = 0 \quad et \quad u_1 = (1, \ldots, 1)^T \tag{3.64}$$

$$0 = \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n \tag{3.65}$$

Le nombre de valeurs propres nulles correspond au nombre de sous-graphes disjoints.

Le nombre de valeurs propres faibles ($\lambda_2 \approx \ldots \approx \lambda_n \approx 0$) correspond au nombre moins 1 de sous-graphes quasi-disjoints.

La diagonalisation de la matrice L de l'équation 3.62 est

$$L = P^{-1} \Lambda P \tag{3.66}$$

avec Λ la matrice diagonale des valeurs propres et P la matrice de passage qui contient en colonnes les vecteurs propres associés aux valeurs propres.

Dans le cas de notre exemple donné avec le graphe 3.20, on a :

$$\Lambda \approx \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0.07 & 0 & 0 & 0 \\ 0 & 0 & 0.43 & 0 & 0 \\ 0 & 0 & 0 & 0.55 & 0 \\ 0 & 0 & 0 & 0 & 1.84 \end{pmatrix} \tag{3.67}$$

et

$$P \approx \begin{pmatrix} 0.45 & -0.54 & -0.04 & 0.72 & -0.02 \\ 0.45 & -0.55 & -0.09 & -0.70 & -0.01 \\ 0.45 & 0.28 & 0.55 & -0.02 & 0.64 \\ 0.45 & 0.35 & 0.33 & -0.02 & -0.75 \\ 0.45 & 0.45 & -0.76 & 0.02 & 0.15 \end{pmatrix} \tag{3.68}$$

Comme indiqué dans la formule 3.64, nous constatons que la première valeur propre est nulle et que le premier espace propre est $Vect(1, \ldots, 1)$. Nous pouvons remarquer que la première valeur propre non nulle λ_2 est très petite. Ce qui signifie que le graphe est composé de deux sous-graphes quasi-disjoints. Il est aussi à noter que le vecteur propre associé u_2 a ses deux premières composantes négatives et les trois autres positives. Ce qui signifie que les deux sous-graphes quasi-disjoints sont $\{a; b\}$ et $\{c; d; e\}$.

3.4.1.6 Matrices laplaciennes normalisées

Deux autres formes de matrices laplaciennes peuvent être considérées : les matrices laplaciennes normalisées.

La matrice laplacienne normalisée « random walk » est la matrice de transfert des marches aléatoires [Mohar, 1997]. Elle contient les probabilités de passage d'un nœud à un autre calculées en fonction des poids des liens. Elle est donnée par la formule

$$L_{rw} \quad = \quad D^{-1}\, L \quad = \quad I \; - \; D^{-1}\, W \tag{3.69}$$

La matrice laplacienne normalisée « random walk » du graphe 3.20 est

$$L_{rw} \approx \begin{pmatrix} 0.231 & -0.192 & -0.038 & 0 & 0 \\ -0.194 & 0.225 & -0.031 & 0 & 0 \\ -0.026 & -0.021 & 0.471 & -0.423 & 0 \\ 0 & 0 & -0.381 & 0.524 & -0.143 \\ 0 & 0 & 0 & -0.231 & 0.231 \end{pmatrix} \tag{3.70}$$

La matrice laplacienne normalisée symétrique est

$$L_{sym} \quad = \quad D^{-1/2}\, L\, D^{-1/2} \quad = \quad I \; - \; D^{-1/2}\, W\, D^{-1/2} \tag{3.71}$$

La matrice laplacienne normalisée symétrique du graphe 3.20 est

$$L_{sym} \approx \begin{pmatrix} 0.231 & -0.193 & -0.032 & 0 & 0 \\ -0.193 & 0.225 & -0.026 & 0 & 0 \\ -0.032 & -0.026 & 0.471 & -0.402 & 0 \\ 0 & 0 & -0.402 & 0.524 & -0.182 \\ 0 & 0 & 0 & -0.182 & 0.231 \end{pmatrix} \tag{3.72}$$

L'intérêt de la matrice L_{sym} est que nous avons une matrice symétrique qui permet d'utiliser des algorithmes de calcul des valeurs propres beaucoup plus rapides qu'avec la matrice L_{rw} qui n'est pas symétrique.

Ces différentes matrices laplaciennes ont un spectre qui conduit à des coupes qui peuvent être légèrement différentes. Considérons le graphe coupé en k sous-graphes $A_1, \cdots A_k$ avec $|A_i|$

le nombre de nœuds du graphe A_i et $vol(A_i)$ la somme des poids du graphe A_i. Le laplacien non normalisé conduit à des coupes qui minimisent le *RatioCut* [von Luxburg, 2007] :

$$RatioCut(A_1, \cdots, A_k) = \sum_{i=1}^{k} \frac{cut\left(A_i, \overline{A_i}\right)}{|A_i|} \tag{3.73}$$

Les laplaciens normalisés conduisent à des coupes qui minimisent le *Ncut* [von Luxburg, 2007] :

$$Ncut(A_1, \cdots, A_k) = \sum_{i=1}^{k} \frac{cut\left(A_i, \overline{A_i}\right)}{vol(A_i)} \tag{3.74}$$

Ces quelques bases de la théorie spectrale des graphes étant présentées, voici maintenant au paragraphe 3.4.2 une présentation du Clustering Spectral.

3.4.2 Le Clustering Spectral automatique

Luxburg présente le Clustering Spectral dans son tutoriel [von Luxburg, 2007].

Soit $X = (x_i)_{i \in [\![1,n]\!]}$ l'ensemble des n données que l'on veut partitionner en K classes. Les algorithmes du Clustering Spectral se décomposent en 3 étapes :

1. un graphe de similarité est d'abord construit entre les objets ;
2. une projection est effectuée sur un espace spectral où les clusters sont plus facilement identifiables ;
3. pour finir, un clustering convexe standard est effectué sur les données dans cet espace spectral.

Ces trois étapes sont présentées dans la suite.

3.4.2.1 Étape 1 : la construction du graphe de similarité

La construction du graphe de similarité peut être séparée en deux temps : *la construction des liens* suivie de *la pondération des liens* [von Luxburg, 2007].

- En général, *la construction des liens* est faite selon l'une des approches suivantes :
  1. le graphe des ε-voisinages qui relie entre eux les objets distants de moins de ε. Cependant cette méthode a un inconvénient car il faut ajuster le paramètre ε manuellement ;
  2. le graphe des k-plus proches voisins qui peut être rendu non orienté avec une matrice d'adjacence symétrique. Cette symétrie peut être obtenue en utilisant l'une des deux procédures suivantes :
     - la procédure de k-plus proches voisins *symétriques* consiste à retenir tous les liens entre les nœuds i et j dont i est l'un des k plus proches voisins de j **ou** dont j est l'un des k plus proches voisins de i ;
     - la procédure de k-plus proches voisins *mutuels* consiste à retenir tous les liens entre les nœuds i et j dont i est l'un des k plus proches voisins de j **et** dont j est l'un des k plus proches voisins de i.

     Le paramètre k doit être fixé a priori. Il est préconisé de le fixer à une valeur proche de $log(n)$ où n est le nombre de nœuds [von Luxburg, 2007] ;

3. le graphe obtenu par une combinaison des deux approches précédentes en mixant le ε-graphe avec le k-NN graphe ;
4. le graphe totalement connecté qui présente l'inconvénient d'être plus coûteux pour la suite du Clustering Spectral que les graphes obtenus avec les approches précédentes.

Dans la pratique, le k-NN graphe est souvent choisi en premier choix avec $k \approx \log(n)$ et adapté si les résultats ne sont pas satifaisants [von Luxburg, 2007].

- En ce qui concerne *la pondération des liens*, la procédure suivante est employée. Une pondération $s(x_i, x_j)$ est assignée à chaque lien construit entre les objets x_i et x_j. Cette pondération est généralement normalisée dans l'intervalle $[0, 1]$. Les pondérations peuvent être :
  1. une similarité binaire : $s(x_i, x_j) = 1$ s'il existe un lien entre x_i et x_j et 0 sinon. Dans ce cas, on parle aussi de graphe non pondéré ;
  2. une similarité gaussienne : $s(x_i, x_j) = \exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right)$ avec σ le paramètre contrôlant la largeur des voisinages ;
  3. toute autre pondération qui définit une mesure de similarité sur l'ensemble des données.

  La similarité binaire est intéressante car elle ne nécessite aucun ajustement de paramètre. Avec un paramètre σ bien ajusté, la similarité gaussienne peut donner de meilleurs résultats. Cependant avec un σ mal ajusté, les résultats sont facilement dégradés.
  La similarité s définie ici conduit à la définition d'une matrice d'adjacence W, avec $w_{ij} = s(x_i, x_j)$. W peut être sparse si la construction du graphe limite le nombre de liens. Ceci permet des gains de temps de calcul conséquents avec des algorithmes adaptés [von Luxburg, 2007].

3.4.2.2 Étape 2 : la construction de l'espace spectral

Définissons maintenant la matrice diagonale des degrés D donnée par l'équation 3.57. Ceci nous permet de définir la matrice laplacienne avec trois variantes couramment utilisées :

1. le laplacien non normalisé L de l'équation :

$$L = D - W \tag{3.75}$$

2. le laplacien normalisé « marche aléatoire » L_{rw} de l'équation :

$$L_{rw} = D^{-1}L = I - D^{-1}W \tag{3.76}$$

3. le laplacien normalisé symétrique L_{sym} de l'équation :

$$L_{sym} = D^{-1/2}LD^{-1/2} = I - D^{-1/2}WD^{-1/2} \tag{3.77}$$

Dans la pratique, les trois laplaciens sont tous les trois utilisés. L'algorithme populaire de Shi et Malik [Shi et Malik, 2000] correspond à une utilisation du laplacien normalisé symétrique.

Ensuite, les K premiers vecteurs propres associés aux K plus petites valeurs propres du laplacien sont calculés et disposés dans la matrice $V \in \mathbb{R}^{n \times K}$. V correspond à une projection des similitudes dans un espace propre de plus petite dimension où les clusters sont supposés être plus faciles à identifier. D'un point de vue coupe de graphe, les K dimensions de V fournissent les K premières coupes binaires de plus faible connectivité.

3.4.2.3 Étape 3 : le partitionnement des données dans l'espace spectral

Après les deux étapes précédentes, les éventuelles variétés caractérisées par une connectivité complexe sont censées avoir été dépliées dans un espace propre. Une méthode de clustering convexe classique est maintenant capable d'identifier ces clusters. Les méthodes de l'état de l'art utilisent généralement un simple k-means [von Luxburg, 2007]. Cependant, d'autres méthodes de clustering convexe peuvent également être utilisées, comme par exemple les mixtures de gaussiennes [Xiong *et al.*, 2014].

3.4.2.4 Remarque

Dans les structurations par projection, nous avons détaillé quelques méthodes et évoqué d'autres. Il existe une méthode de projection qui correspond aux deux premières étapes du Clustering Spectral. Il s'agit du Laplacian Eigenmaps (LE) aussi nommée *Spectral Embedding* qui est une technique projective basée sur une décomposition spectrale de la matrice du laplacien [Belkin et Niyogi, 2003]. Elle permet de projeter l'espace des données initiales à N dimensions sur l'espace spectral à $K < N$ dimensions correspondant aux K plus petites valeurs propres.

3.5 Bilan

Au paragraphe 3.1 nous avons détaillé les différentes natures que peuvent revêtir les données et comment produire des mesures entre ces données. Ces considérations sont le préalable à l'analyse des données. Ceci est examiné dans toutes nos réalisations et particulièrement dans la première partie du chapitre 4 où nous sélectionnons et fusionnons des descripteurs hétérogènes.

Ensuite aux paragraphes 3.2 et 3.3, nous avons présenté les techniques de structuration par projection et classification. Ces structurations nous intéressent dans toutes nos réalisations que ce soit au chapitre 4 où nous développons une méthode originale de classification par corrélation, comme au chapitre 5 où nous implémentons et étudions différentes techniques de Clustering Spectral supervisé ou semi-supervisé.

Le Clustering Spectral présenté au paragraphe 3.4 est la technique de classification sur laquelle nous nous focalisons au chapitre 5.

Tout au long de cet état de l'art, nous avons aussi présenté des critères pour évaluer les résultats des techniques de structuration. Nous les employons dans toutes nos réalisations pour l'évaluation des résultats.

Chapitre 4

Mesures de ressemblances et corrélation

Résumé : Dans ce chapitre nous nous intéressons aux mesures de ressemblances entre vidéos avec des techniques de corrélation. Nous abordons dans la première partie la sélection de descripteurs et leur fusion. Nous commençons par examiner comment produire une vérité terrain par paire sur une base de vidéos. Nous continuons par étudier l'extraction des descripteurs de différentes natures. Puis nous présentons notre technique originale de sélection et de fusion des descripteurs. Cette méthode est basée sur l'utilisation des coefficients de corrélation de rang. Ceux-ci nous permettent de nous affranchir des ordres de grandeurs en ne retenant que les ordonnancements. Ils nous permettent ainsi de produire une mesure de ressemblance automatique conforme à ce qu'un opérateur humain peut qualifier de « ressemblant ». Dans la dernière partie de ce chapitre, nous présentons une technique originale de classification basée elle aussi sur les coefficients de corrélation de rang.

Dans beaucoup de domaines, la quantité de données multimédia augmentant fortement, il est nécessaire de disposer d'outils de recherche et de navigation adaptés. Nous avons vu ces outils de visualisation au chapitre 2. Dans les besoins de structuration, nous avons vu qu'il est souvent nécessaire de commencer par produire des mesures de similarités entre les données multimédia. Beaucoup de travaux existent sur les images fixes [Datta *et al.*, 2008]. Pour les vidéos, la quantité de données est beaucoup plus grande et l'analyse est beaucoup plus complexe. Comme pour les images statiques, les descripteurs extraits des vidéos sont de plus bas niveau que ce qui est attendu par les utilisateurs. C'est ce que l'on appelle le fossé sémantique. L'objectif du travail envisagé est de développer des méthodologies permettant de mesurer des similarités entre films. Les données exploitées sont à la fois des descripteurs de bas niveau et les informations textuelles de type métadonnée accompagnant les films.

Nous avons donc besoin d'explorer différentes directions. Pour quantifier les ressemblances, il convient d'examiner les problèmes d'extraction de descripteurs et de mesure de distance. Pour regrouper les films qui se ressemblent et envisager une structuration de la base de données, il faut investiguer les techniques de classification. Nous nous référons au chapitre 3 pour avoir une vision d'ensemble sur ces problématiques. Enfin, pour pouvoir effectuer les deux tâches précédentes sur des données de type texte et image, il convient d'effectuer des fusions de données de natures hétérogènes.

Autour de la mesure de similarité, des défis tels que TRECVID [Smeaton *et al.*, 2006] avec des tâches telles que l'indexation sémantique, la détection de copie basée sur le contenu ou MediaEval [Ionescu *et al.*, 2012] avec la classification automatique par genre donnent une bonne vision des domaines de recherche. Toutes ces tâches sont généralement basées sur des mesures de similarité ou de dissimilarité. Cependant les méthodes sont en général basées sur la classification mais la contrainte forte est d'avoir une base entièrement annotée et de taille suffisante pour permettre une généralisation. Nous avons donc été amenés à examiner comment produire cette mesure grâce à la fusion des informations que l'on peut extraire. Notre travail [Voiron *et al.*, 2012] qui utilise une partie de la base vidéo de la CITIA présentée au paragraphe 4.1 consiste en une investigation sur l'apport des informations textuelles de type métadonnée ajoutées à des informations bas niveau. Nous quantifions cet apport et proposons une stratégie de fusion adaptée à notre problématique. Nous réutilisons les résultats du travail que Benoit [Benoit *et al.*, 2011] a déjà fait sur la fusion des informations bas niveau à l'aide de l'intégrale de Choquet sur les mêmes données. La figure 4.1 présente un extrait de la base de la CITIA avec une relation de ressemblance donnée par un expert par rapport à la vidéo centrale.

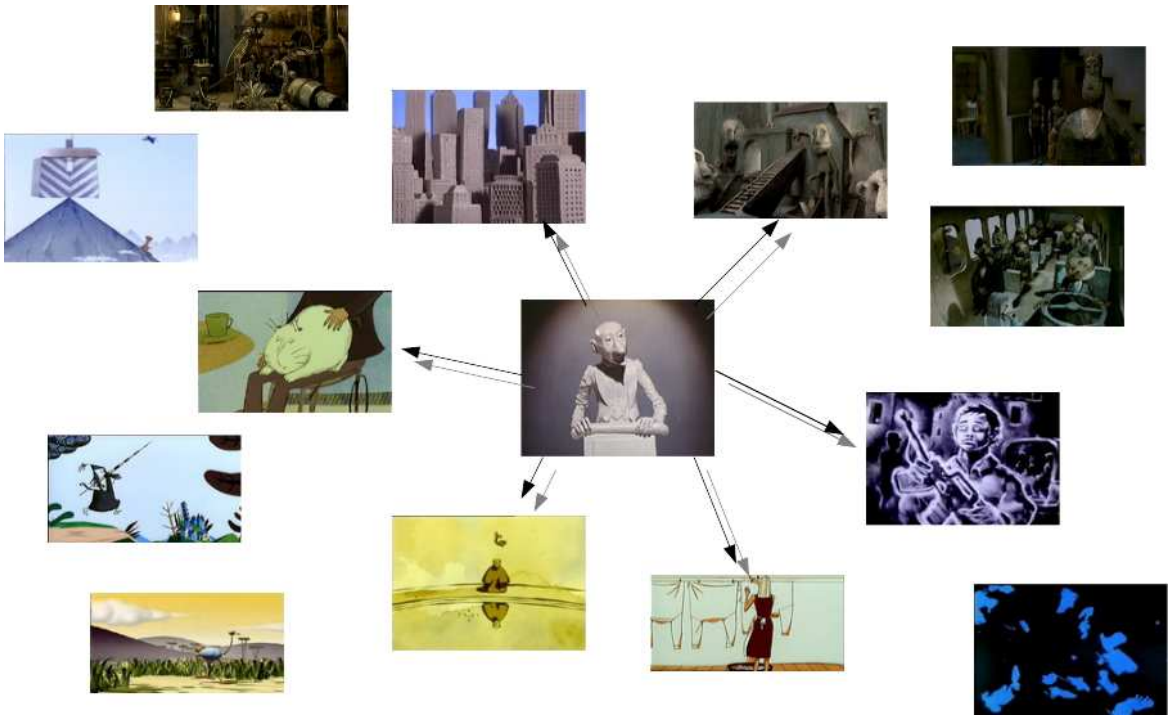


FIGURE 4.1 – Extrait de la base CITIA avec une relation de ressemblance donnée par un expert.

Pour une base documentaire vidéo, nous n'avons pas directement une mesure de distance. Cependant, nous savons extraire de nombreuses caractéristiques bas niveau et très souvent, les vidéos sont fournies avec des informations textuelles de type métadonnée (titre, durée, réalisateur, producteur...). Avec ces données, nous détaillons au paragraphe 4.2 comment produire des mesures de dissimilarité. Il convient ensuite de comparer toutes ces informations pour sélectionner celles qui sont les plus pertinentes. Pour pouvoir effectuer ces comparaisons, les indices de corrélation de rang sont détaillés au paragraphe 4.3. Ensuite, la sélection de descripteurs est présentée au paragraphe 4.4. Puis au paragraphe 4.5, nous développons une

technique originale de fusion pour obtenir une unique mesure proche des attentes de l'utilisateur. Nous avons effectué ce travail avec un extrait de la base de la CITIA. L'originalité de ce travail repose sur l'utilisation des coefficients de corrélation de rang avec des informations issues de vidéos. Une autre contribution porte sur une méthode de tris successifs pour produire une mesure fusionnée.

Dans un dernier paragraphe 4.6, nous réemployons ces coefficients de corrélation de rang pour mettre au point une méthode originale de classification. Ces coefficients permettent de s'affranchir des ordres de grandeur et nous permettent de classifier directement des données non normalisées. Nous verrons aussi les autres caractéristiques de cette méthode de classification. Nous l'appliquerons à la tâche de classification par genre sur les données vidéo du challenge MediaEval [Schmiedeke *et al.*, 2012].

4.1 La base de la CITIA et sa vérité terrain par paires

Ce chapitre décrit des méthodes évaluées sur une base de données liée au contexte local.

4.1.1 Présentation générale

La base de la CITIA (figure 4.2), est composée de films présentés aux différentes éditions du Marché International du Film d'Animation d'Annecy (http://www.annecy.org/mifa/presentation:fr). Pour cette base, nous disposons des informations textuelles de type métadonnée accompa-

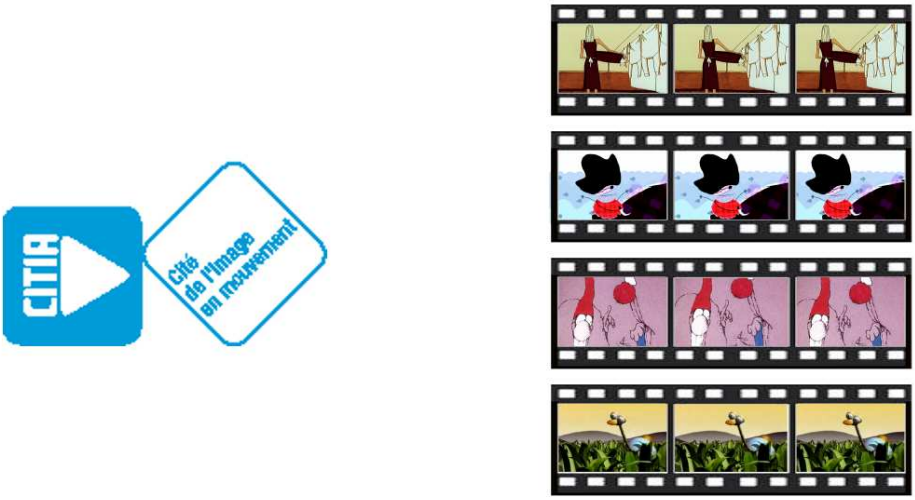


FIGURE 4.2 – CITIA, cité de l'image en mouvement (http://www.citia.org).

gnant les films (titre, année, durée, pays, public, genre, catégorie, réalisateur, producteur, technique, synopsis anglais et français). Un extrait de la base CITIA est donné avec ses informations textuelles de type métadonnée dans le tableau 4.1.

Cette base ne fournit pas de vérité terrain correspondant au problème de mesure de ressemblance que nous posons. Il existe différents descripteurs textuels de type métadonnée mais aucun n'a la portée d'un classement de type « vérité terrain ». Si l'on se plaçait dans le cadre d'une reconnaissance par genre, le genre deviendrait notre vérité terrain. Il en serait de

Titre original	Année	Durée	Pays	Public
Casa	2003	07'07"	France	12-15_ans Jeunes_adultes Adultes
Circuit marine	2003	07'50"	France Canada	Tous_public
David	1977	08'45"	Pays-Bas	Tous_public
Gazoon	1998	03'30"	France	Tous_public

Genre	Catégorie	Technique	Réalisateur
Artistique Dramatique	Court-métrage	Dessin_sur_cellulos	Sylvie_Léonard
Aventure	Court-métrage	Dessin_sur_papier ...	Isabelle_Favez
Comique	Court-métrage	Dessin_sur_cellulos	Paul_Driessen
Artistique	Fin_d'étude	Animation_3D	Romain_Villemaine

Producteur	Synopsis anglais (après lemmatisation)
Folimage Arte	Casablanca summer woman live return young_man ...
Folimage ONF_Canada	eat not be question
Cine_Cartoon_Centre	eternal battle big small here main character never ...
Sparx_Animation_Studios	facetious bird torment ostrich help friend elephant

TABLE 4.1 – Exemple de données textuelles de type métadonnée pour quatre films de la base de la CITIA.

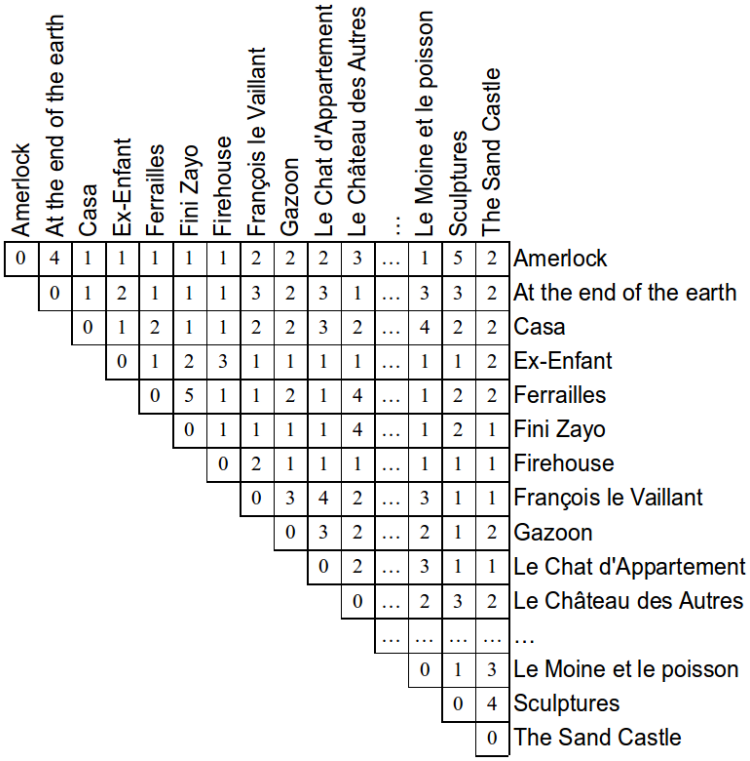


FIGURE 4.3 – La matrice de la vérité terrain sous forme de similarité entre les 51 films de notre base d'expérimentation.

même pour la technique ou tout autre descripteur textuel de type métadonnée. Cependant les experts de ce domaine ne peuvent actuellement s'accorder sur une ontologie sur ces données. La base de la CITIA nous a toutefois intéressé car la vérité terrain que nous pouvions produire naturellement est une ressemblance entre vidéos. Cette notion de ressemblance est une des préoccupations importante dans l'exploration visuelle. Très souvent lors d'une recherche, nous progressons de proche en proche en sélectionnant des média ressemblants pour aboutir à celui que l'on cherche précisément. Cette notion de ressemblance, bien que subjective, a réellement un sens pour les films d'animation de la base de la CITIA qui sont des films de courte durée avec une certaine homogénéité.

4.1.2 Obtention d'une vérité terrain

Nous avons quantifié la ressemblance entre films de la façon suivante. Nous avons construit une vérité terrain mesurant la ressemblance entre chaque paire de films sur une échelle de 5 valeurs ordonnées allant de « très ressemblant » à « très différent ». Trois opérateurs humains ont effectué une annotation manuelle dont le détail est décrit dans [Benoit *et al.*, 2011]. Les trois opérateurs ont attribué un degré de similarité sur une échelle de notation allant de 1 à 5 (très ressemblant, ressemblant, neutre, différent, très différent). Ceci pour chacune des 1275 paires différentes formées avec les 51 films d'animation retenus. On obtient donc une matrice d'ordre 51, symétrique dont les éléments diagonaux sont forcés à 0. Ensuite, pour chaque paire de film, nous retenons une valeur unique. Sur ce jeu de données annoté par peu d'experts, nous avons fusionné les annotations par la médiane et la moyenne des trois observateurs humains.

- La médiane choisit toujours la valeur la plus consensuelle. La médiane donne une matrice qui prend toujours les 5 mêmes valeurs allant de 1 à 5. La figure 4.3 présente cette matrice symétrique sous forme de similarité.
- La moyenne produit une dissimilarité discrète composée de 14 valeurs différentes. C'est une vérité terrain moins discrète mais plus sensible au bruit. Les résultats des expérimentations de sélection et fusion de descripteurs par corrélation avec la moyenne l'ont montré en donnant des résultats moins concluants. Le bruit pourrait expliquer les faibles valeurs des coefficients de concordance obtenus.

Dans la suite, nous utilisons la médiane.

Une application Web (figure 4.4) a été développée pour recueillir ces vérités terrains. Un serveur web envoie à l'interface web deux vidéos de la base. L'opérateur peut les visionner en parallèle et doit évaluer leur ressemblance à l'aide de l'échelle à 5 niveaux décrite précédemment. Afin de limiter les phénomènes d'adaptation de l'utilisateur, le serveur web propose les vidéos de manière aléatoire. Plusieurs opérateurs peuvent travailler en parallèle. Dans ce cas, pour obtenir plus rapidement une annotation complète le serveur évite dans un premier temps de proposer les mêmes paires. L'aspect collaboratif de cette application devient intéressant lorsque le nombre d'opérateurs augmente. Lorsque les vérités terrain des opérateurs se recouvrent, le serveur peut effectuer différents traitements comme le calcul de la vérité terrain médiane ou moyenne. Le fait d'avoir plusieurs avis sur les mêmes paires permet d'avoir une qualité d'annotation plus fine. La gestion des utilisateurs et des évaluations permet aux opérateurs d'arrêter leur comparaisons pour les reprendre ultérieurement. Pour gagner du temps, l'opérateur n'est pas obligé de visionner les films en entier à chaque comparaison. On peut noter que le protocole est très souple et manque de rigueur. Cependant dans notre contexte, il satisfait bien l'objectif d'obtention rapide des vérités terrain.

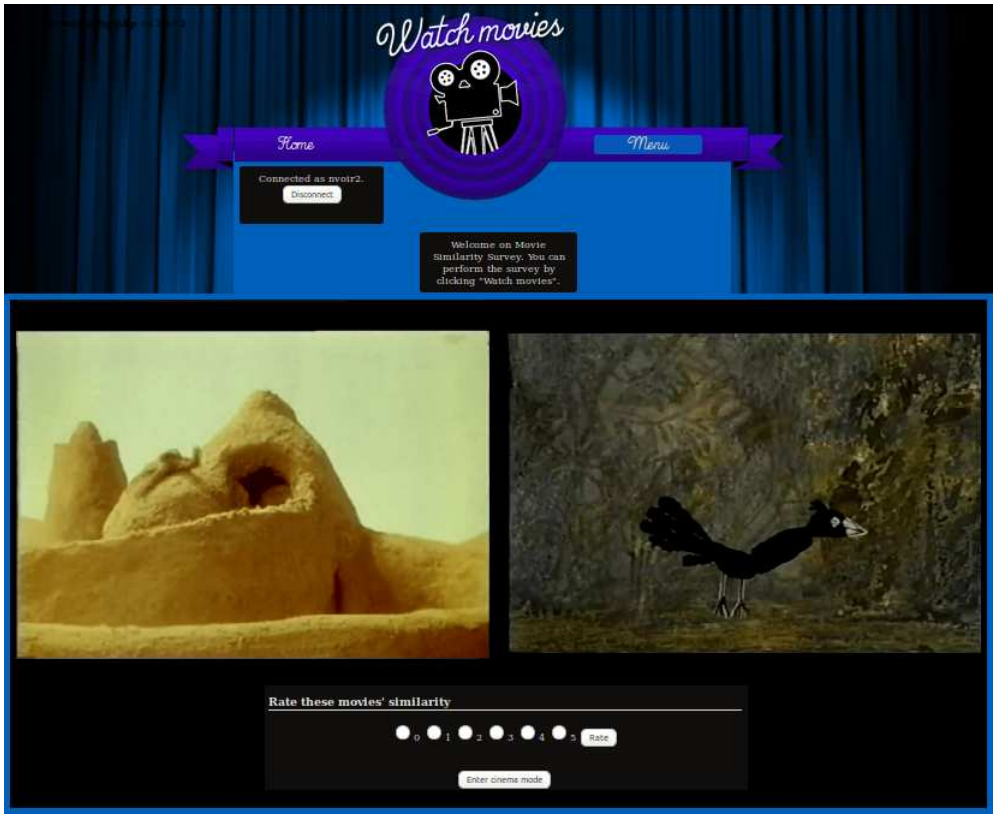


FIGURE 4.4 – Application web « Movie Survey » : construction collaborative de similarité par paires de vidéos avec un extrait de la base de la CITIA.

Pour la suite, notre travail est fortement conditionné par cette vérité terrain qui est donnée par une variable ordinale, ce qui nous a amené à travailler sur les paires de films et à ordonner leurs ressemblances avec les mesures de dissimilarité décrites ci-après.

4.2 Des données aux mesures de dissimilarités

Les dissimilarités, normalisées, propres, ainsi que les distances sont définies au chapitre 3. Avec les données de la base CITIA, nous construisons des dissimilarités normalisées propres qui ne sont pas forcément des distances. Les dissimilarités normalisées sont intéressantes car elles ont la même amplitude en prenant toutes leurs valeurs entre 0 et 1. Les dissimilarités propres présentent l'avantage de séparer les données car deux données différentes ont une valeur de dissimilarité strictement positive. La valeur de dissimilarité nulle permet d'identifier que l'on a affaire à deux fois la même donnée. Nous n'avons pas besoin que les dissimilarités soient forcément des distances car l'inégalité triangulaire n'est pas utile dans les méthodes que nous allons développer dans ce chapitre. Nous disposons alors des dissimilarités normalisées propres suivantes :

(i) Les premières dissimilarités sont celles issues de l'annotation humaine décrite au paragraphe 4.1. La matrice d'ordre 51 est recodée avec cinq valeurs croissantes différentes (de $0,1$ associée à « très ressemblant », à $0,9$ associée à « très différent »). 0 est volontairement exclu pour satisfaire le critère de séparabilité des dissimilarités propres (équation

3.7). On a donc bien une dissimilarité discrète qui prend toujours les 5 mêmes valeurs allant de $0,1$ à $0,9$.

(ii) La deuxième famille de dissimilarités est composée de deux distances euclidiennes normalisées décrites dans [Benoit *et al.*, 2011]. Elles sont adaptées aux films d'animation. Elles sont issues de la fusion de descripteurs bas niveau extraits à partir des caractéristiques de couleur et rythme estimés sur la globalité de chaque film [Ionescu, 2007]. Là encore, l'homogénéité d'un même film, en terme de rythme ou de couleur, donne du sens à ces descripteurs globaux. La première des deux distances est une moyenne pondérée. La seconde est basée sur l'intégrale de Choquet [Grabisch *et al.*, 2008] qui contrairement à une simple moyenne pondérée tient aussi compte des interactions entre les caractéristiques. Dans les deux cas, les poids sont ajustés grâce à une étape d'apprentissage en utilisant l'annotation humaine. Plus de détails sont donnés dans le papier [Benoit *et al.*, 2011].

(iii) La troisième famille est composée des *dissimilarités textuelles* obtenues depuis les métadonnées qui accompagnent les films. Un exemple d'informations textuelles de type métadonnée associées à 4 films est présenté dans le tableau 4.1. Pour chacune de ces métadonnées, nous avons construit une mesure de dissimilarité :

(a) Tous les films ont été produits au cours des 50 dernières années. Ainsi, avec l'équation (4.1), nous proposons une mesure de dissimilarité normalisée fondée sur l'information « Année » où $year(x)$ est l'année de réalisation du film x.

$$d_{year}(x,y) = \frac{|year(x) - year(y)|}{50} \tag{4.1}$$

Le tableau 4.2 affiche dans sa deuxième colonne, cette dissimilarité pour les 4 films donnés dans le tableau 4.1.

(b) La plupart des films durent moins de 20 minutes. Ainsi avec l'équation (4.2), nous proposons une mesure de dissimilarité normalisée entre 2 films x et y fondée sur l'information « Durée » quantifiée en secondes.

$$d_{dur}(x,y) = \min\left(1\ ,\ \frac{|dur(x) - dur(y)|}{1200}\right) \tag{4.2}$$

Le tableau 4.2 affiche dans sa troisième colonne, cette dissimilarité pour les 4 films donnés dans le tableau 4.1.

Couple de films	d_{year}	d_{dur}	d_{ctry}	d_{gnr}
Casa / Circuit marine	0	0,036	0,5	0
Casa / David	0,52	0,082	1	0
Casa / Gazoon	0,1	0,181	0	0,5
Circuit marine / David	0,52	0,046	1	0
Circuit marine / Gazoon	0,1	0,217	0,5	0
David / Gazoon	0,42	0,263	1	0

TABLE 4.2 – Extrait des dissimilarités textuelles de type métadonnée entre quatre films de la base CITIA pour quatre critères : année de production, durée, pays et genre.

(c) Pour un film x, toutes les autres données textuelles de type métadonnée sont décrites par un ensemble L_x qui est une liste de mots ou de mots-clés. Par exemple,

« Dessins sur cellulos » et « Dessins sur papier » sont considérés comme deux mots-clés différents.

Avec l'équation (4.3), nous proposons d'utiliser la dissimilarité normalisée dérivée du classique indice de Jaccard décrit dans l'état de l'art du chapitre 3.

$$d_{\cdot}(x,y) = 1 - \frac{|L_x \cap L_y|}{|L_x \cup L_y|} \tag{4.3}$$

Le tableau 4.2 illustre ces dissimilarités pour les informations « Pays » et « Genre ».

Il existe d'autres indices que celui de Jaccard : par exemple l'indice de Braun-Blanquet décrit dans l'état de l'art du chapitre 3. Nous avons expérimenté la dissimilarité normalisée obtenue à partir de l'indice de Braun-Blanquet (équation (4.4)).

$$d_{\cdot}(x,y) = 1 - \frac{|L_x \cap L_y|}{\max\left(|L_x|, |L_y|\right)} \tag{4.4}$$

Mais aucune amélioration n'a été notée par rapport à l'indice de Jaccard.

(d) Dans le cas particulier des synopsis qui sont des textes bien plus longs que ceux examinés ci-dessus, nous avons suivi la même approche. Mais nous avons au préalable converti tous les mots par leurs lemme/racine en utilisant un logiciel de lemmatisation [1]. La ponctuation et la répétition des mots ont été ignorées. Des exemples sont donnés dans le tableau 4.1 avec les synopsis anglais. Ainsi pour le synopsis « A facetious bird torments an ostrich with the help of his friend, the elephant. », nous obtenons la forme lemmatisé « facetious bird torment ostrich help friend elephant ».

(iv) La quatrième famille est composée de deux *dissimilarités de référence* obtenues par une distribution aléatoire. La première est continue, la seconde est discrète. Les valeurs sont prises aléatoirement de manière uniforme respectivement dans l'intervalle $[0;1]$ et l'ensemble $\{0;0,5;1\}$. Ces distributions permettent de quantifier la performance des différentes dissimilarités construites à partir des différents descripteurs par rapport à la référence donnée par les dissimilarités construites aléatoirement.

4.3 La corrélation de rang

Avec les dissimilarités définies au paragraphe 4.2, deux questions peuvent être posées :

— Est-ce que les mesures de dissimilarité calculées sont proches des mesures de dissimilarité des perceptions humaines ?

— Peut-on combiner tout ou partie de ces dissimilarités pour modéliser plus finement la perception humaine ?

Nous répondrons à ces deux questions aux paragraphes 4.4 et 4.5. Mais avant cela, nous avons besoin d'une solution pour comparer et fusionner les dissimilarités. En effet, les approches numériques simples telles que les moyennes de dissimilarités échouent car les distributions des dissimilarités en question peuvent être très différentes. C'est ce que l'on peut constater avec les histogrammes de quatre dissimilarités présentées dans la figure 4.5. Les dissimilarités issues des descripteurs « année » et « durée » ont des valeurs presque continues et étalées entre 0 et 1 tandis que celles issues des descripteurs « genre » et « public » sont

1. http ://www.sphinx-soft.com

fortement discrètes avec uniquement 5 valeurs différentes. En outre, les moyennes des quatre dissimilarités sont respectivement 0,29 ; 0,18 ; 0,89 et 0,25 avec des écarts-types de 0,22 ; 0,15 ; 0,23 et 0,43. Ces valeurs sont fortement différentes et caractérisent des séries statistiques bien différentes. Ainsi, pour résoudre ce problème d'hétérogénéité des caractéristiques statistiques des dissimilarités, nous proposons de considérer uniquement les rangs des valeurs des dissimilarités en utilisant une approche basée sur les coefficients de corrélation de rang.

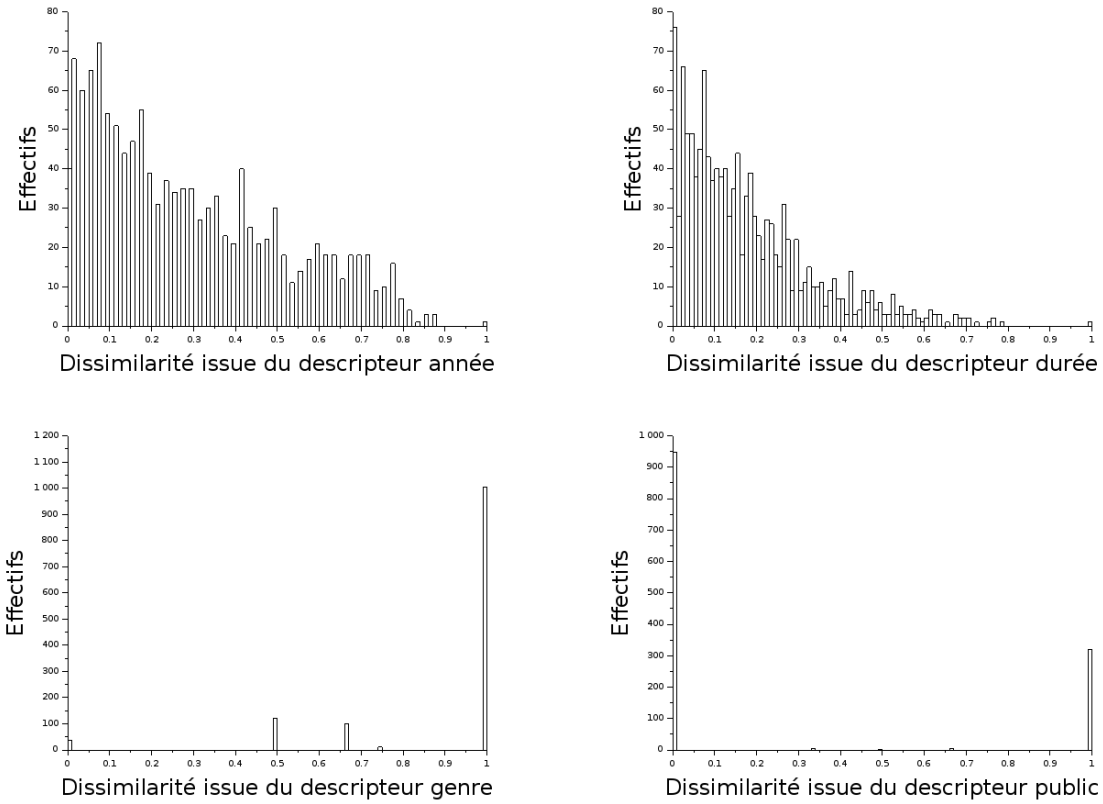


FIGURE 4.5 – Histogrammes en 100 classes donnant les effectifs des dissimilarités issues des descripteurs « année », « durée », « genre » et « public »

4.3.1 Le tau de Kendall

En considérant un ensemble de n objets $E = \{x_1, ..., x_n\}$ et deux mesures de dissimilarité d_1, d_2, la première façon de comparer les classements induits est le concept d'équivalence. Une façon classique de définir cette équivalence est la suivante : si d_1 et d_2 sont continues, alors elles sont dites *équivalentes en ordre* si et seulement si $\forall\, i, j, k, l \in \mathbb{R}^4$ tels que $1 \leqslant i, j, k, l \leqslant n$

$$d_1(x_i, x_j) < d_1(x_k, x_l) \Leftrightarrow d_2(x_i, x_j) < d_2(x_k, x_l) \tag{4.5}$$

Cependant, étant donné que tous les quadruplets doivent vérifier l'équation (4.5), cette définition est trop stricte. Pour relâcher cette contrainte comme dans « Fuzzy order-equivalence for similarity measures »[Rifqi *et al.*, 2008], nous pouvons étendre cette définition en mesurant un degré d'équivalence basé sur l'utilisation du tau de Kendall [Kendall et Gibbons, 1990].

Ce coefficient est donné par l'équation(4.6) :

$$\tau = \frac{C_4 - D_4}{N_4} \tag{4.6}$$

où N_4 est le nombre de quadruplets (x_i, x_j, x_k, x_l) composés de paires différentes (x_i, x_j) et (x_k, x_l), elles-mêmes composées d'objets différents. Dans ce contexte, C_4 et D_4 sont les nombres de quadruplets concordants et discordants. Ces deux derniers concepts d'accord sont définis comme suit. Étant données deux mesures de dissimilarité d_1 et d_2, nous disons qu'un quadruplet (x_i, x_j, x_k, x_l) est :

(i) *concordant* si

$$\begin{cases} d_1(x_i, x_j) < d_1(x_k, x_l) \\ d_2(x_i, x_j) < d_2(x_k, x_l) \end{cases} \quad ou \quad \begin{cases} d_1(x_i, x_j) > d_1(x_k, x_l) \\ d_2(x_i, x_j) > d_2(x_k, x_l) \end{cases} \tag{4.7}$$

(ii) *discordant* si

$$\begin{cases} d_1(x_i, x_j) > d_1(x_k, x_l) \\ d_2(x_i, x_j) < d_2(x_k, x_l) \end{cases} \quad ou \quad \begin{cases} d_1(x_i, x_j) < d_1(x_k, x_l) \\ d_2(x_i, x_j) > d_2(x_k, x_l) \end{cases} \tag{4.8}$$

(iii) lié (ni concordant, ni discordant) si

$$d_1(x_i, x_j) = d_1(x_k, x_l) \quad ou \quad d_2(x_i, x_j) = d_2(x_k, x_l) \tag{4.9}$$

Comme les distances sont symétriques, si le quadruplet (x_i, x_j, x_k, x_l) est concordant (respectivement discordant), alors les quadruplets (x_j, x_i, x_k, x_l), (x_i, x_j, x_l, x_k) et (x_j, x_i, x_l, x_k) sont aussi concordants (respectivement discordants).

On peut également noter que si le quadruplet (x_i, x_j, x_k, x_l) est concordant (respectivement discordant), alors le quadruplet (x_k, x_l, x_i, x_j) est aussi concordant (respectivement discordant).

Ainsi pour réduire les temps de calcul, lors du dénombrement de concordances et discordances, l'examen des quadruplets peut être limité à (x_i, x_j, x_k, x_l) avec $j > i$ et $((k = i$ et $l > j$) ou ($k > i$ et $l > k))$. Dans ce contexte, le nombre N_4 doit être diminué au nombre de quadruplets examinés :

$$N_4 = \frac{\left(\frac{n(n-1)}{2}\right)\left(\frac{n(n-1)}{2} - 1\right)}{2} = \frac{(n+1)n(n-1)(n-2)}{8} \tag{4.10}$$

Les considérations précédentes nous permettent de mettre en place l'algorithme 1 qui permet d'examiner tous les quadruplets nécessaires.

Si le tau de Kendall prend sa valeur maximale +1 alors les deux dissimilarités sont équivalentes en ordre.

Toutefois, ce coefficient qui compare les dissimilarités sur tous les quadruplets est trop global et dépasse notre objectif. Nous voulons seulement un moyen de classer les dissimilarités entre les différents média de façon à proposer les voisins les plus ressemblants. Nous sommes donc à la recherche d'un classement des objets par rapport à un objet central donné. Cette cible sera bien sûr amenée à se déplacer avec la navigation effectuée par l'utilisateur. Nous n'avons donc pas besoin d'un classement qui inclut les quadruplets formés de paires non connectées et alors notre objectif correspond seulement à l'examen des triplets d'objets.

Comme dans [Hubert et Arabie, 1985], considérons les triplets d'objets à la place des quadruplets. Considérons toujours le même ensemble de n objets $E = \{x_1, ..., x_n\}$ et les mêmes mesures de dissimilarité d_1 et d_2. Un triplet (x_i, x_j, x_k) est :

Algorithm 1 - Algorithme d'examen des quadruplets nécessaires.

```
for i = 1 to n do
  for j = i + 1 to n do
    k ← i
    for l = j + 1 to n do
      Examen du quadruplet (x_i, x_j, x_k, x_l)
    end for
    for k = i + 1 to n do
      for l = k + 1 to n do
        Examen du quadruplet (x_i, x_j, x_k, x_l)
      end for
    end for
  end for
end for
```

(i) *concordant* si :

$$\begin{cases} d_1(x_i, x_j) < d_1(x_i, x_k) \\ d_2(x_i, x_j) < d_2(x_i, x_k) \end{cases} \quad \text{ou} \quad \begin{cases} d_1(x_i, x_j) > d_1(x_i, x_k) \\ d_2(x_i, x_j) > d_2(x_i, x_k) \end{cases} \tag{4.11}$$

(ii) *discordant* si :

$$\begin{cases} d_1(x_i, x_j) > d_1(x_i, x_k) \\ d_2(x_i, x_j) < d_2(x_i, x_k) \end{cases} \quad \text{ou} \quad \begin{cases} d_1(x_i, x_j) < d_1(x_i, x_k) \\ d_2(x_i, x_j) > d_2(x_i, x_k) \end{cases} \tag{4.12}$$

(iii) *lié* (ni concordant,ni discordant) si :

$$d_1(x_i, x_j) = d_1(x_i, x_k) \quad ou \quad d_2(x_i, x_j) = d_2(x_i, x_k) \tag{4.13}$$

On peut noter que les triplets (x_i, x_j, x_k) et (x_i, x_k, x_j) ont le même comportement d'accord. Ainsi, pour réduire les temps de calcul, l'examen des triplets peut être limité à (x_i, x_j, x_k) avec $j \neq i$, $k \neq i$ et $j < k$. Dans ce contexte, le nombre de triplets à considérer est :

$$N_3 = \frac{n(n-1)(n-2)}{2} \tag{4.14}$$

Le *tau de Kendall* sur les triplets est :

$$\tau = \frac{C_3 - D_3}{N_3} \tag{4.15}$$

où C_3 et D_3 sont les nombres de triplets concordants et discordants sur les N_3 triplets considérés. Le tau de Kendall prend ses valeurs entre -1 et 1. Un coefficient nul signifie que les deux dissimilarités sont indépendantes. Un coefficient égal à 1 (respectivement -1) signifie que le classement des dissimilarités est le même (respectivement opposé). Plus généralement, plus le tau de Kendall est proche de 1, meilleur est l'accord entre les deux dissimilarités.

4.3.2 Le gamma de Goodman-Kruskal et l'indice de discrétion

Si l'une des deux dissimilarités est fortement discrète, le tau de Kendall peut être positif, proche de zéro alors que les concordances peuvent être fortement plus nombreuses que

les discordances. L'explication provient du grand nombre de valeurs liées par la dissimilarité fortement discrète. En conséquence, pour les dissimilarités discrètes nous avons besoin d'un indice insensible au nombre de triplets liés. Nous proposons l'utilisation du *gamma de Goodman et Kruskal* décrit par Podani [Podani, 1997] :

$$\gamma = \frac{C_3 - D_3}{C_3 + D_3} \tag{4.16}$$

Pour des dissimilarités continues, il est égal au tau de Kendall. Sa seule différence de comportement est de ne pas prendre en compte les triplets liés. Avec ces deux indices, nous pouvons en déduire un troisième, quantifiant globalement le caractère discret des deux dissimilarités. Il s'agit du *pourcentage de triplets non liés* :

$$\pi = \frac{\tau}{\gamma} = \frac{C_3 + D_3}{N_3} \tag{4.17}$$

Avec ces trois indices, nous pouvons comparer des dissimilarités variées. En particulier, l'utilisation du gamma de Goodman et Kruskal avec la dissimilarité issue de la vérité terrain permet de quantifier et comparer la qualité de tous les ordonnancements partiels des dissimilarités discrètes.

4.4 Sélection de descripteurs

4.4.1 Comparaisons des dissimilarités

Nous voulons répondre à la première question posée au début du paragraphe 4.3. À savoir si les mesures de dissimilarité calculées sont proches des mesures de dissimilarité des perceptions humaines. Pour cela, on peut utiliser directement les indicateurs définis précédemment pour sélectionner les descripteurs les plus corrélés à la vérité terrain sur un ensemble de test pour lequel nous disposons d'annotations humaines.

La figure 4.6 présente le tau de Kendall sur triplets entre l'annotation humaine et 13 mesures de dissimilarité extraites automatiquement depuis les descripteurs des documents. Les 11 dissimilarités textuelles sont présentes. Les 2 autres dissimilarités, appelées « Choquet » et « Weighted », proviennent des données bas niveau (voir le paragraphe 4.2).

Le point le plus frappant est que la « Technique » est la meilleure information bien loin devant toutes les autres mesures de dissimilarité. Donc, si une seule dissimilarité doit être choisie pour simuler notre comportement humain, la « Technique » est la plus appropriée. Les informations suivantes les plus performantes sont les deux dissimilarités de bas niveau. Le tau de Kendall corrobore les comparaisons de Benoit [Benoit *et al.*, 2011] : « la fusion avec l'intégrale de Choquet est meilleure que la somme pondérée ». Toutes les autres caractéristiques sont des dissimilarités textuelles et donnent des tau de Kendall moins élevés, voir proche de zéro pour les dernières. Cela peut s'expliquer par le caractère discret des dissimilarités textuelles et la continuité des autres. Toutefois, ces dissimilarités ne sont pas nécessairement hors d'intérêt. Pour poursuivre en excluant les valeurs liées, nous devons regarder le deuxième indice.

La figure 4.7 montre le gamma de Goodman et Kruskal sur les triplets entre la dissimilarité humaine et les mêmes 13 mesures de dissimilarité présentées dans la figure 4.6. Dans un premier temps, nous remarquons que la « Technique » est toujours la meilleure. En sachant

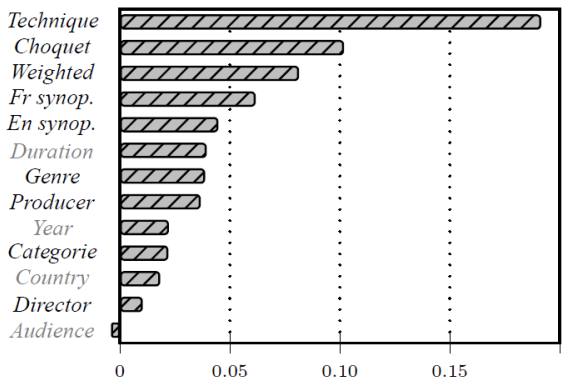


FIGURE 4.6 – Tau de Kendall pour chacun des descripteurs extraits sur la base CITIA.

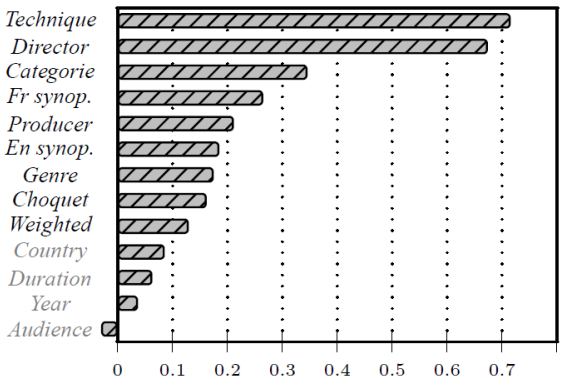


FIGURE 4.7 – Gamma de Goodman et Kruskal pour chacun des descripteurs extraits sur la base CITIA.

que $\gamma_{\ Technique} \approx 0,716$ et en résolvant l'équation (4.16), nous trouvons que $C_3 = 6,04 \times D_3$. Ceci signifie que les concordances avec l'opinion humaine sont plus de six fois plus nombreuses que les discordances. En sachant que $\tau_{\ Technique} \approx 0,192$, avec l'équation (4.17) nous obtenons $\pi_{\ Technique} \approx 27\%$. Ceci signifie que seulement 27% des paires de films ont été ordonnées par la Technique avec 23% de concordances et 4% de discordances.

En comparaison, pour la dissimilarité obtenue avec l'intégrale de Choquet, $\tau_{\ Choquet} \approx 0,102$, $\gamma_{\ Choquet} \approx 0,162$ et nous obtenons $\pi_{\ Choquet} \approx 63\%$. Ainsi les concordances sont $1,4$ fois plus nombreuses que les discordances. Comme la dissimilarité obtenue avec l'intégrale de Choquet est continue, elle se compose de valeurs distinctes. Donc les triplets ne peuvent pas être liés par la dissimilarité obtenue avec l'intégrale de Choquet, mais plutôt par la dissimilarité issue de l'annotation humaine. Par conséquent, 63% est le pourcentage de triplets non liés par l'annotation humaine.

Ainsi, si l'on compare l'écart entre les 27% obtenus en utilisant seulement la « Technique » avec ces 63%, ceci nous indique qu'il y a 36% des paires qui sont liées et qui pourraient être déliées par d'autres descripteurs. En d'autres termes, cela signifie que ces 36% représente notre marge de progression dans une stratégie de fusion de descripteurs. Il faut donc trouver d'autres critères (différents de la « Technique ») à prendre en compte pour augmenter le nombre de concordances.

Un autre point est que le $\gamma_{\ Director}$ est presque égal au $\gamma_{\ Technique}$, mais $\tau_{\ Director}$ est très faible. Ceci indique que la dissimilarité « Réalisateur » avec laquelle est calculée $\tau_{\ Director}$ est un bon critère pour le classement. Toutefois, en raison des nombreuses valeurs liées, il ne classe que peu de paires. Or, dans la base de données utilisée, pour chaque film, il y a un seul réalisateur et parfois deux co-réalisateurs, qui sont presque toujours différents. Cela signifie que les observateurs humains classent les films du même réalisateur plus similaires que ceux de réalisateurs différents. Grâce au fait que les réalisateurs utilisent souvent les mêmes techniques dans leurs différents films, une dépendance entre « Réalisateur » et « Technique » pourrait exister. Ceci est une supposition. Mais à ce point, nous ne pouvons pas la prouver. C'est ce que nous ferons au paragraphe 4.5 où la méthode proposée du *gamma restant* montrera si les concordances du « Réalisateur » sont totalement, partiellement ou non incluses dans les concordances de la « Technique ».

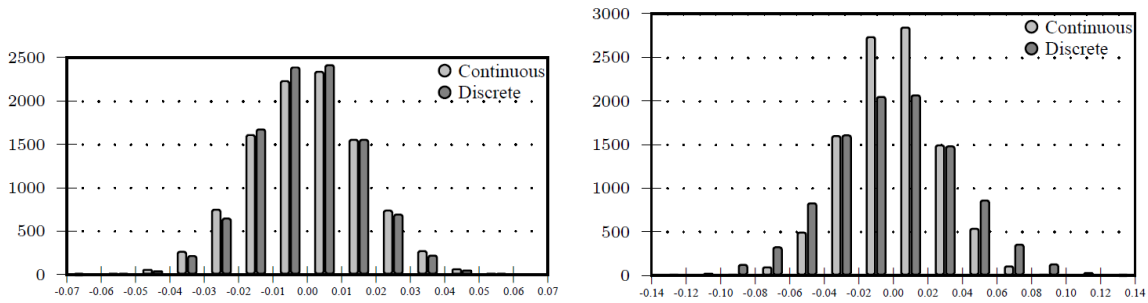
Enfin, de la même manière, les gamma de Goodman et Kruskal pour les dissimilarités

« Catégorie », « Producteur », « Genre », « Synopsis français et anglais » sont plus élevés que celui des dissimilarités de bas niveau. Mais des interdépendances pourraient aussi apparaître. Les synopsis français et anglais pourraient être liés. Il en est de même pour « Producteur » et « Réalisateur », « Synopsis » et « Genre »...

4.4.2 Comparaisons avec l'aléatoire

Enfin, afin de mieux cerner les dissimilarités « Pays », « Durée », « Année » et « Public » qui ont des petites valeurs pour leur tau de Kendall et leur gamma de Goodman-Kruskal, nous les comparons avec les dissimilarités aléatoires à valeurs dans l'intervalle $[0;1]$. La première est continue et la seconde est discrète. Leurs tau de Kendall et gamma de Goodman-Kruskal ont été calculés sur 10 000 échantillons aléatoires.

Une lecture de la figure 4.8 nous indique que les distributions ont des allures gaussiennes. Moins de $0,5\%$ des dissimilarités aléatoires ont un tau de Kendall en dehors de l'intervalle $[-0,05;0,05]$ et un gamma de Goodman-Kruskal en dehors de l'intervalle $[-0,1;0,1]$. En conséquence, les figures 4.6 et 4.7 présentent en gris clair, les quatre mesures de dissimilarité, qui restent dans le périmètre des comportements aléatoires. En particulier, les indices de l'information « Public » sont proches zéro et à valeurs négatives. Ce critère est une information subjective qui peut être composé d'entrées non-homogènes fournies par différents observateurs humains. Ainsi, l'information « Public » n'est pas une donnée pertinente pour notre travail. Les informations de « Durée » et d'« Année » sont des informations objectives, mais apparemment non utilisables (tout au moins pour l'ensemble de films que nous avons considéré). C'est assez surprenant car intuitivement nous nous serions attendus à ce que des films de la même période soient plus similaires entre eux que des films d'autres périodes. Cela signifie également qu'il n'y a aucun lien entre l'année et d'autres informations comme la technique. De même, le pays n'est pas non plus une information intéressante. Cependant, les films considérés ne sont pas géographiquement hétérogènes. De plus, la dissimilarité que nous avons utilisée n'est pas une distance géographique ou culturelle. Elle est binaire et indique simplement si les films proviennent du même pays ou non. Cependant ces remarques sont à relativiser par rapport à la taille restreinte de l'échantillon composé uniquement de 51 films. Un ensemble de données plus large serait nécessaire pour obtenir des conclusions plus solides.



(a) Tau de Kendall.

(b) Gamma de Goodman et Kruskal.

FIGURE 4.8 – Fréquence des coefficients de corrélation de rang avec 10 000 dissimilarités continues aléatoires et 10 000 dissimilarités discrètes aléatoires.

4.5 Fusion de descripteurs

4.5.1 La méthode de fusion par tri successif

Nous voulons maintenant répondre à la deuxième question posée au début du paragraphe 4.3. À savoir si l'on peut combiner tout ou partie des dissimilarités pour modéliser plus finement la perception humaine.

Le problème est de construire une dissimilarité qui fusionne toutes les dissimilarités, en sachant que certaines sont discrètes et d'autres continues. La solution que nous proposons est l'utilisation d'une approche par tri lexicographique.

Soient deux ensembles ordonnés A et B. L'ordre lexicographique sur le produit cartésien $A \times B$ est défini par $(a, b) \leq (a', b')$ si et seulement si $(a < a')$ *ou* $((a = a')$ *et* $b \leq b')$.

La première étape de cette approche est d'identifier un ordre entre les dissimilarités (comme l'ordre littéral dans un alphabet). Nous proposons une méthode basée sur le gamma de Goodman et Kruskal que l'on va appeler le *gamma restant*. Soit d_{hu} l'annotation humaine et d_1, ..., d_p les p autres mesures de dissimilarité. Dans un premier temps, les p gamma de Goodman et Kruskal entre d_{hu} et d_i sont calculés. La plus grande valeur de ces gamma est sélectionnée car elle identifie la dissimilarité qui contient le plus grande proportion de concordances sur le nombre de paires non liées. La dissimilarité associée $d_{(1)}$ est placée en tête . Elle est utilisée pour classer toutes les paires pour lesquelles c'est possible. Ensuite, nous calculons les $p - 1$ *gamma restants* qui sont les gamma de Goodman et Kruskal entre d_{hu} et d_i en ne considérant que les paires encore liées (qui n'ont pas été ordonnées avec la dissimilarité $d_{(1)}$). Cette technique est appliquée de manière itérative jusqu'à ce qu'il n'y ait plus de dissimilarité à fusionner ou qu'il n'y ait plus de valeur liée.

On peut noter que, comme nous utilisons un ordre lexicographique, le tri successif est arrêté dès qu'une dissimilarité continue est utilisée. Ceci provient du fait qu'une dissimilarité continue permet de trier toutes les paires : la probabilité d'avoir une distance égale à 0 entre deux films est nulle. Après l'identification d'un ordre spécifique entre les dissimilarités, nous classons tous les couples de films selon cet ordre lexicographique. Cette stratégie est appliquée itérativement jusqu'à ce que toutes les dissimilarités soient utilisées ou jusqu'à ce qu'un classement total (sans ex-aequo) soit obtenu. Et finalement, la dissimilarité fusionnée est obtenue par normalisation en divisant tous les rangs de classement par le nombre total de paires.

Un exemple de ce processus est donné dans le tableau 4.3 avec les 6 paires des 4 films présentés dans les tableaux 4.1 et 4.2. Dans cet exemple, on suppose que le gamma restant donne l'ordre : 1.« Pays/d_{ctry} », 2.« Année/d_{year} » puis 3.« Durée/d_{dur} ». La colonne « rang » est le rang obtenu en appliquant l'ordre lexicographique de la façon suivante :

- Le premier des 6 couples est le seul pour lequel $d_{ctry} = 0$. Son rang est 1.
- Ensuite, les deuxième et troisième couples ont tous les deux $d_{ctry} = 0,5$.
- Ils sont séparés par le second critère d_{year}. Ainsi leurs rangs sont respectivement 2 et 3, selon l'ordre des valeurs qu'ils prennent pour d_{year} .
- Et ainsi de suite jusqu'au sixième et dernier couple.
- Et pour finir $d_f = \dfrac{rang}{6}$ nous fournit une dissimilarité normalisée propre.

Couple de films	d_{ctry}	d_{year}	d_{dur}	$rang$	d_f
Casa / Gazoon	0	.	.	1	0,167
Casa / Circuit marine	0,5	0	.	2	0,333
Circuit marine / Gazoon	0,5	0,1	.	3	0,5
David / Gazoon	1	0,42	.	4	0,667
Circuit marine / David	1	0,52	0,046	5	0,833
Casa / David	1	0,52	0,082	6	1

TABLE 4.3 – Fusion par tris successifs en utilisant l'ordre lexicographique d_{ctry}, d_{year}, d_{dur}.

4.5.2 Résultats

La figure 4.9 montre l'évolution du *gamma restant* tout au long des différentes étapes du tri successif sur l'ensemble de la base de données CITIA considérée. Dans la configuration proposée, six étapes sont nécessaires pour converger. Ce schéma synthétise la totalité du processus de la méthode. Il s'interprète de la manière suivante. Initialement, à l'étape zéro (abscisse 0), les *gamma restants* (en ordonnée) sont exactement les gamma de Goodman et Kruskal de la figure 4.7 pour les douze dissimilarités considérées. La « Technique » présente la meilleure valeur. Ensuite, pour passer à l'étape une, toutes les paires sont triées selon les valeurs de la dissimilarité « Technique ». Le gamma est calculé sur les triplets qui sont encore liés afin d'identifier les critères les plus importants à utiliser lors de l'étape de tri. Le résultat est visible à l'abscisse 1 : le deuxième meilleur résultat provient de l'information « Réalisateur ». Et ainsi de suite jusqu'à rencontrer la première dissimilarité continue : « Choquet » à l'étape 5. Et un *gamma restant* nul est obtenu à l'étape 6, pour toutes les autres dissimilarités.

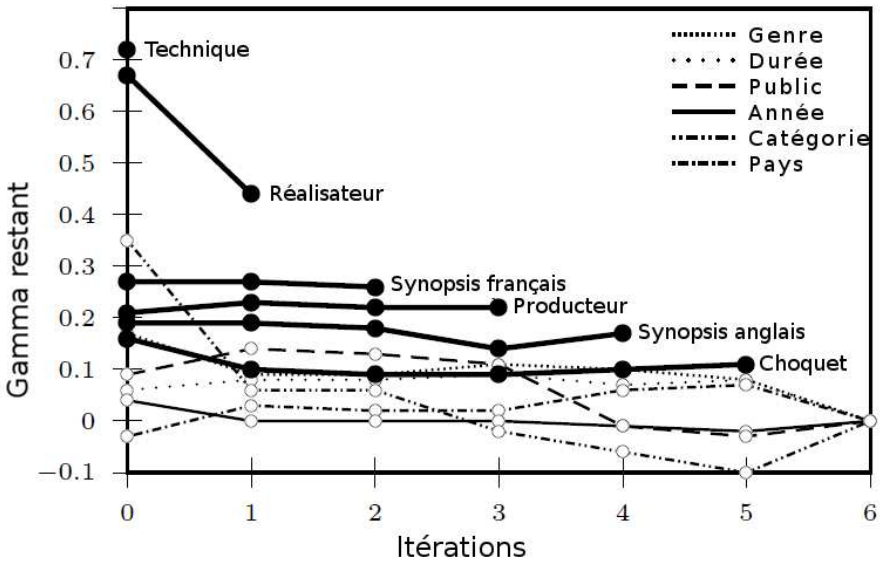


FIGURE 4.9 – Évolution du *gamma restant* tout au long des étapes du tri successif. Les lignes continues épaisses correspondent aux dissimilarités utilisées. Les autres lignes, en traits fins, correspondent aux dissimilarités non utilisées.

Le premier résultat visible est que le gamma du « Réalisateur » diminue fortement à l'étape 1. Cela semble confirmer que l'information du « Réalisateur » est en partie liée à celle

de la « Technique ». Cependant, « Réalisateur » est toujours le deuxième critère de tri. Il peut également être noté que la « Catégorie » diminue fortement et n'est ensuite plus pertinente du tout. Cette observation démontre l'intérêt de la méthode *gamma restant* pour exclure les critères de tri redondants. Ensuite, les dissimilarités « Synopsis français », « Producteur », « Synopsis anglais » et « Choquet » sont utilisées successivement. Avec ses valeurs continues, toutes différentes, « Choquet » clos nécessairement le processus. « Catégorie » et cinq autres dissimilarités représentées en traits fins ne sont donc pas exploitées.

Au paragraphe 4.4.1, on a vu que pour le critère « Technique » utilisé seul : $\tau_{\ Technique} \approx 0,192$ et $\gamma_{\ Technique} \approx 0,716$. On a aussi vu dans ce paragraphe 4.4.1 que l'on avait une marge de progression de 36% entre les $27\% = \dfrac{\tau_{\ Technique}}{\gamma_{\ Technique}}$ de paires classées en utilisant seulement la « Technique » et les 63% de triplets non liés par l'annotation humaine.

Notre processus de fusion nous permet d'obtenir un tau de Kendall : $\tau_{\ Fusion} \approx 0,260$ avec un Gamma de Goodman et Kruskal : $\gamma_{\ Fusion} \approx 0,414$ avec 63% de paires classées. Ce 63% est le maximum que l'on peut atteindre car c'est le nombre de paires non liées par l'annotation humaine. De la définition du Gamma de Goodman et Kruskal 4.16, nous déduisons qu'il y a $\dfrac{1+0,414}{1-0,414} \approx 2,413$ fois plus de concordances que de discordances. Et comme la somme des concordances et des discordances représentent les 63% de paires non liées, on déduit qu'il y a $44,5\%$ de concordances et $18,5\%$ de discordances sur l'effectif global.

Donc par rapport au critère « Technique » utilisé seul, ce processus de fusion permet d'améliorer les résultats dans les proportions suivantes :

- Premièrement, les valeurs liées sont fortement diminuées, passant de $73\% = 100\% - 27\%$ à $37\% = 100\% - 63\%$.
- Deuxièmement, les concordances avec l'opinion humaine sont augmentées de façon significative. Elles passent 23% à $44,5\%$. Cependant les discordances sont elles aussi augmentées. Mais dans un volume un peu plus faible. Elles passent de 4% à $18,5\%$.

En conclusion, la fusion proposée permet d'ordonner presque la moitié ($44,5\%$) de la base de données comme un humain le ferait.

On aborde maintenant l'aspect coût de calcul. Comme le montre la figure 4.9, 12 dissimilarités ont été utilisées. Lors des différentes étapes, nous calculons 12 gamma, puis 11, puis 10 et ainsi de suite jusqu'à la fin du tri successif. Donc dans le cas le plus défavorable, un maximum de $12 + 11 + 10 + \cdots + 1 = 78$ gamma pourraient être calculés. Cependant, dans notre contexte, la figure 4.9 montre que notre méthode du *gamma restant* a calculé seulement $12 + 11 + 10 + \cdots + 6 = 63$ gamma.

Pour connaître la pertinence de notre méthode, nous avons recherché le meilleur tri existant pour pouvoir comparer nos résultats. En envisageant l'approche grossière basée sur le calcul systématique de tous les gamma des tris possibles, trouver le meilleur tri avec ces 12 dissimilarités nous amène à considérer les $12! = 479\,001\,600$ tris possibles. Après avoir considéré ces 479 001 600 tris et calculé leurs 479 001 600 gamma, nous obtenons seulement 57 tris plus performants que le résultat obtenu par notre méthode du *gamma restant*. Autrement dit, la méthode du *gamma restant* nous permet de trouver le $58^{ème}$ meilleur tri sur 479 001 600 en calculant seulement 63 gamma au lieu des 479 001 600 gamma à calculer que nécessite la recherche systématique du meilleur gamma. Ce qui représente un gain de temps de calcul d'un rapport de 1 pour $\dfrac{479\,001\,600}{63} = 7\,603\,200$.

Pour comparer l'amélioration apportée par notre méthode avec le meilleur tri possible, la

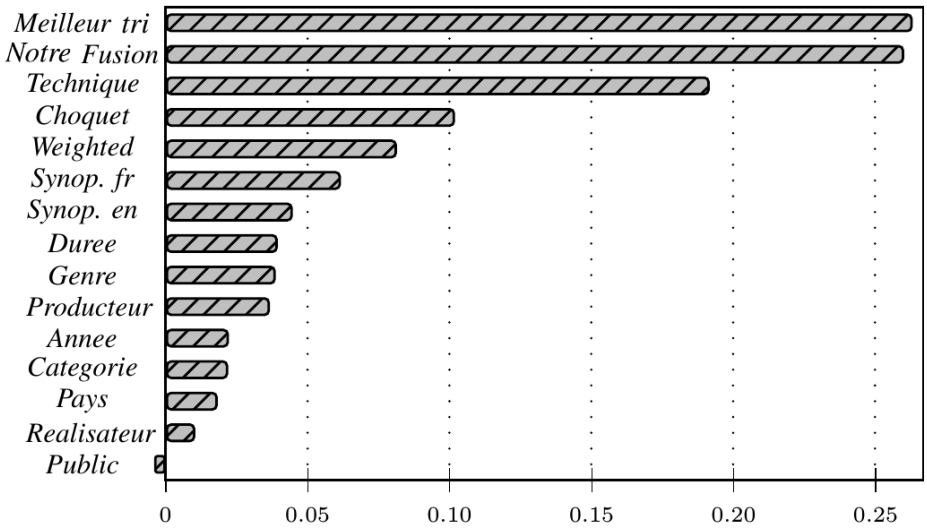


FIGURE 4.10 – Résultat de la fusion : le tau de Kendall pour chacun des descripteurs extraits sur la base CITIA ainsi que ceux de la fusion par Gamma restant et du meilleur tri possible.

figure 4.10 reprend les tau de Kendall :

- des 12 dissimilarités,
- de notre méthode du *gamma restant* ($\tau_{Fusion} \approx 0,260$),
- du meilleur tri obtenu par un examen exhaustif ($\tau_{Meilleur} \approx 0,263$).

On constate que notre méthode de sélection de descripteur permet d'obtenir une bien meilleure qualité que chaque descripteur pris séparément, en s'approchant fortement du résultat du meilleur tri possible. $\tau_{Meilleur} \approx 0,263$ présente une valeur supérieure au résultat de notre méthode : $\tau_{Fusion} \approx 0,260$ de seulement $\frac{0,263 - 0,260}{0,263} \approx 1\%$.

Ce 1 pour 100 de perte de performance par rapport au 1 pour 7 603 200 de gain de temps met en évidence l'intérêt du *gamma restant* en terme de « Qualité » par rapport au « Temps de calcul ».

4.5.3 Améliorations envisagées

Une brève exploration des possibilités d'amélioration de la méthode des tris successifs a été réalisée. La première amélioration est basée sur l'idée qu'un classement selon une dissimilarité peut être découpé en 2 sous-classements. Afin d'expliquer cette idée, prenons par exemple, la technique qui est graduelle, de 0 signifiant que les deux films utilisent exactement les mêmes techniques, à 1 indiquant que toutes leurs techniques sont différentes. Toutes les valeurs comprises entre 0 et 1 indiquent la proportion de techniques identiques. Une dissimilarité binaire partielle appelée « Technique2 » peut être créée comme étant la partie entière de la valeur de la technique. Ses valeurs sont 0 pour « utilise au moins une technique commune » et 1 pour « pas de technique en commun ». Considérons l'exemple des films du tableau 4.4. Les colonnes 2 et 3 du tableau 4.5 présente les dissimilarités « Technique », « Technique2 ».

Nous voyons sur cet exemple que les 2 dissimilarités « Technique » et « Technique2 » n'ont pas les mêmes valeurs. Le classement selon la dissimilarité partielle « Technique2 » est

Titre original	Technique	Réalisateur
Ferrailles	Marionnettes Animation_d'objets	Laurent_Pouvaret
Petite escapade	Marionnettes	Pierre-Luc_Granjon
La cancion du microsillon	Marionnettes	Laurent_Pouvaret

TABLE 4.4 – 3 films avec leurs techniques et réalisateurs.

Couple de films	d_{tech}	d_{tech2}	d_{prod}	rg_1	rg_2
Ferrailles / Petite escapade	0,5	0	1	3	3
Ferrailles / La cancion du microsillon	0,5	0	0	2	1
Petite escapade / La cancion du microsillon	0	0	1	1	2

TABLE 4.5 – Les dissimilarités « Technique », « Technique2 », « Réalisateur » avec les 2 classements selon les tris successifs (« Technique » puis « Réalisateur ») et (« Technique 2 », « Réalisateur » puis « Technique »).

inclus dans le classement selon la dissimilarité « Technique ». Comme nous pouvons le voir dans les 2 dernières colonnes du tableau 4.5, nous pouvons créer 2 classements différents :

— rg_1 est le classement obtenu par le tri suivant « Technique » puis « Réalisateur »,
— rg_2 est le classement obtenu en appliquant « Technique 2 », « Réalisateur » puis « Technique ».

Les dissimilarités fusionnées nommées « Fusion1 » et « Fusion2 » sont obtenues à partir des classements rg_1 et rg_2. En utilisant toujours les 51 mêmes films décrits au début de ce chapitre, les valeurs des gamma de Goodman et Kruskal obtenus sont $\gamma_{Fusion1} \approx 0.587$ et $\gamma_{Fusion2} \approx 0,592$. Cela montre que l'utilisation des dissimilarités partielles peut apporter des améliorations sans nécessiter l'utilisation d'informations supplémentaires. Cependant, un grand nombre de dissimilarités partielles peuvent être construites, de sorte que le nombre de combinaisons possibles explose.

4.5.4 Évaluation de la méthode à l'aide d'une validation croisée

Pour évaluer cette fusion, l'ensemble d'apprentissage est séparé de l'ensemble de validation. 34 films sont prélevés au hasard parmi les 51 films pour former l'ensemble d'apprentissage. Tandis que les 17 restants sont utilisés pour la validation. L'ordre lexicographique est obtenu sur l'ensemble d'apprentissage en utilisant la méthode du *gamma restant*. La mesure de dissimilarité fusionnée est calculée. Afin de pouvoir comparer les résultats, les tau de Kendall sont calculés sur l'ensemble d'apprentissage comme sur l'ensemble de validation.

Pour obtenir des résultats significatifs, nous avons appliqué cette opération 1 000 fois avec différents sous-ensembles aléatoires. La moyenne des tau de Kendall calculés sur les ensembles d'apprentissage est d'environ $0,294$ avec un écart type de $0,02$. Les mêmes indices sur l'ensemble de validation ont pour valeurs $0,268$ et $0,05$.

On peut donc conclure que sur cet ensemble de validation, le tau de Kendall obtenu est légèrement inférieur à 91% de la valeur obtenue sur l'ensemble d'apprentissage. Ceci montre la robustesse de la méthode et un effet de sur-apprentissage modéré à la vue de la taille de la base utilisée.

4.5.5 Appréciation de la méthode

Nous avons réalisé un prototype de logiciel d'exploration de base vidéo. Il a été alimenté par un extrait de la base de la CITIA (figure 4.11). Ce prototype respecte le processus d'élaboration d'une visualisation mise au point au chapitre 2. Après l'extraction des descripteurs textuels et bas niveau, la structure utilisée est le graphe pondéré des cinq plus proches voisins calculés avec la dissimilarité fusionnée au moyen de la méthode du paragraphe 4.5.1. Cette fusion a été effectuée par corrélation avec une vérité terrain obtenue sur un tiers des vidéos. Cette vérité terrain peut être recueillie à l'aide de l'application présentée au paragraphe 4.1. Ce prototype fournit dans la colonne de gauche une « vue d'ensemble » de toutes les vidéos sous forme de liste, avec au centre une « vue en détail » sous la forme d'une lecture de la vidéo sélectionnée, complétée dans la colonne de droite par un « filtre » automatique proposant les cinq vidéos les plus ressemblantes, avec pour chacune sa valeur de dissimilarité avec la vidéo centrale. Deux types de navigation sont proposées :

- de proche en proche en sélectionnant l'une des cinq plus proches vidéos (colonne de droite),
- globale (colonne de gauche).

Avec sa colonne de droite, ce prototype permet une navigation de proche en proche en cliquant sur l'une des cinq vidéos proposées. Avec sa colonne de gauche, il permet de basculer vers n'importe quelle vidéo de la base. Dans la colonne de droite de ce prototype, les valeurs numériques des dissimilarités entre les cinq vidéos de la colonne de droite et la vidéo centrale sont affichées. Ce logiciel permet de vérifier la cohérence de l'aide à la navigation fournie par cette mesure de dissimilarité.

FIGURE 4.11 – Prototype « Movie Similarity » : exploration de vidéos avec un extrait de la base de la CITIA.

4.5.6 Bilan

Dans cette section, nous avons proposé une nouvelle mesure de dissimilarité automatique entre films reproduisant l'opinion humaine. Cette solution est basée sur une fusion entre les dissimilarités bas niveau et métadonnées. Comme l'agrégation des informations différentes ne peut être réalisée de façon numérique, nous avons proposé une solution originale basée sur les coefficients de corrélation de rang. Les performances montrent que les concordances avec les opinions humaines sont améliorées en utilisant la fusion : de 23% en utilisant seulement les meilleurs critères à 44,5% avec la fusion sur notre base d'expérimentation. La validation croisée nous a permis de valider que la dissimilarité produite classe bien en premier les vidéos les plus ressemblantes et un prototype d'application de navigation de proche en proche a été développé, ouvrant à une future évaluation utilisateur que nous n'avons pas eu le temps de mettre en place. Cette mesure de dissimilarité automatique pourrait être utilisée dans une visualisation pour aider l'utilisateur dans sa sélection de films en lui proposant les films qu'il pourrait apprécier.

4.6 Classification par corrélation

Dans les paragraphes 4.4 et 4.5, nous avons utilisé les coefficients de corrélation de rang pour comparer et fusionner des mesures de dissimilarités issues de descripteurs de natures très différentes. Ces coefficients de corrélation de rang permettent de s'affranchir des ordres de grandeurs en ne considérant que les ordonnancements. Nous voulons maintenant utiliser cette caractéristique pour mettre au point une méthode de classification directement opérationnelle sur des objets définis par des descripteurs numériques hétérogènes non normalisés. Cette méthode présentera aussi les avantages d'être automatique en ne nécessitant pas d'ajustement de paramètre et d'être facilement parallélisable.

4.6.1 Les données du challenge MediaEval

Nous allons tester cette méthode avec les données vidéo de la tâche de détection de genre du challenge MediaEval (figure 4.12). Ce challenge est dédié à l'évaluation de nouveaux

FIGURE 4.12 – MediEval (http://www.multimediaeval.org).

algorithmes travaillant sur des données multimédia. Il donne la possibilité de décrire des documents multimédia par un large panel de modalités comme par exemple la reconnaissance vocale, l'analyse des contenus multimédia, l'analyse audio, les informations fournies par des utilisateurs (étiquettes, tweets), les réponses affectives des utilisateurs, les réseaux sociaux, les coordonnées temporelles et géographiques... [Schmiedeke *et al.*, 2012]

La base de vidéos a été exploitée avec des descripteurs image, son et texte fournis par le LAPI de Bucurest (http://imag.pub.ro). La vérité terrain est disponible par un étiquetage de chacun des films selon l'un des 26 genres (art, comédie, documentaire, éducation, sport...).

4.6.2 Corrélation de rang et partitions

Nous allons voir en quelques étapes comment nous pouvons étendre les coefficients de corrélation de rang définis dans les paragraphes précédents à une partition d'objets caractérisés par des descripteurs numériques. Plus précisément, nous allons définir :

- la corrélation de rang entre une dissimilarité et une partition,
- la corrélation de rang entre une dissimilarité et une classe et
- la corrélation de rang entre un descripteur et une classe.

4.6.2.1 Corrélation de rang entre une dissimilarité et une partition

Au paragraphe 4.4, nous avons vu comment comparer deux dissimilarités à l'aide des coefficients de corrélation de rang. En partant de là, nous allons voir comment mesurer une corrélation entre une dissimilarité et une partition :

— Soient N objets O_n avec $n \in [\![1, N[\![$ qui appartiennent à L classes C_l avec $l \in [\![1; L]\!]$.
— Soit d une dissimilarité entre ces N objets.
— Soit δ une dissimilarité telle que $\delta(O_i, O_j) = 0$ si O_i et O_j appartiennent à la même classe et 1 sinon.
— Réutilisons les définitions de triplets concordants, discordants et liés des équations 4.11, 4.12 et 4.13 et appliquons les aux dissimilarités d et δ. Nous obtenons donc qu'un triplet (x_i, x_j, x_k) tel que $\delta(O_i, O_j) = 0$ et $\delta(O_i, O_k) = 1$ est :
  (i) *concordant* si $d(x_i, x_j) < d(x_i, x_k)$,
  (ii) *discordant* si $d(x_i, x_j) > d(x_i, x_k)$,
  (iii) *lié* si $d(x_i, x_j) = d(x_i, x_k)$.

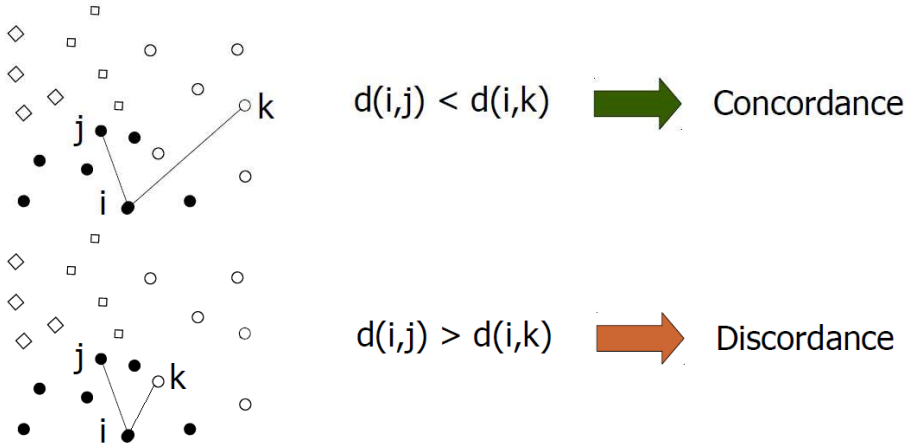


FIGURE 4.13 – Concordance et discordance entre la distance euclidienne sur une répartition bidimensionnelle et une partition en 4 classes (rond plein, rond vide, petit carré et carrés penchés).

La figure 4.13 illustre ces cas de concordances et discordances sur une répartition bidimensionnelle à 4 classes (rond plein, rond vide, petit carré et carré penché). i et j appartiennent à la même classe « rond plein ». k appartient à une autre classe « rond vide ». Dans le premier cas, il y a concordance entre la dissimilarité d et la partition car $d(x_i, x_j) < d(x_i, x_k)$. Dans le second cas, il y a discordance car $d(x_i, x_j) > d(x_i, x_k)$.

— Réemployons la définition du gamma de Goodman et Kruskal : $\gamma_d = \dfrac{C_3 - D_3}{C_3 + D_3}$ (équation 4.16) où C_3 est le nombre de triplets concordants et D_3, le nombre de triplets discordants. Nous avons ainsi défini une corrélation de rang γ_d entre une dissimilarité d et une partition $\bigcup_{l=1}^{L} C_l$.

4.6.2.2 Corrélation de rang entre une dissimilarité et une classe

Si l'on restreint le calcul du gamma de Goodman et Kruskal γ_d du paragraphe 4.6.2.1 aux triplets $(i;j;k)$ tels que $O_i, O_j \in C_l$ et $O_k \notin C_l$, nous définissons un gamma $\gamma_{l,d}$ qui peut être vu soit comme :

— un indice de qualité de la classe C_l basé sur la dissimilarité d, soit inversement,

— un indice indiquant la faculté de la dissimilarité d à séparer la classe C_l.

4.6.2.3 Corrélation de rang entre un descripteur et une classe

Les N objets O_n sont caractérisés par un vecteur de descripteurs numériques $D_{.,p}$ de taille P. On a donc une matrice $D_{n,p}$ avec $n \in [\![1;N]\!]$ et $p \in [\![1;P]\!]$. Pour chacun de ces P descripteurs définissons la dissimilarité d_p telle que $d_p(O_i, O_j) = |D_{i,p} - D_{j,p}|$.

En utilisant le gamma du paragraphe 4.6.2.2 avec la dissimilarité d_p, nous avons une corrélation de rang $\gamma_{l,p}$ entre le $p^{ème}$ descripteur et chaque classe C_l.

4.6.2.4 Interprétation sur des distributions unidimensionnelles

Chaque descripteur correspond à une distribution unidimensionnelle des objets. Suivant ces distributions, le gamma de Goodman et Kruskal $\gamma_{l,d}$ du paragraphe 4.6.2.3 prend des valeurs comprises entre -1 et 1 :

— Une valeur proche de 1 signifie que le $p^{ème}$ descripteur homogénéise bien la classe C_l tout en la séparant bien des autres classes. Dans le détail, ceci signifie que pour ce descripteur, les objets de la classe C_l prennent des valeurs plus proches des autres objets de la même classe que de ceux des autres classes. La première ligne de la figure 4.14, donne un exemple de répartition correspondant à un tel gamma.

FIGURE 4.14 – Trois répartitions unidimensionnelles de la classe C_l (croix rouges) par rapport aux autres classes (ronds bleus) induisant les valeurs extrêmes du gamma de Goodman et Kruskall.

— Une valeur proche de 0 signifie que le $p^{ème}$ descripteur hétérogénéise la classe C_l en ne la séparant pas des autres. C'est le cas d'une répartition aléatoire illustrée en ligne 2 de la figure 4.14.
— Une valeur proche de -1 est peu probable. Ceci signifie que pour le $p^{ème}$ descripteur tous les objets de la classe C_l sont plus éloignés entre eux qu'avec les éléments des autres classes. Un exemple d'une telle répartition est donné en ligne 3 de la figure 4.14.

4.6.3 La méthode de classification par corrélation de rang

Nous avons mis au point une méthode de classification par corrélation de rang basée sur les gamma de Goodman et Kruskall entre les descripteurs et les classes. Lors de la phase d'apprentissage, nous calculons les gamma entre tous les descripteurs et toutes les classes. Lors de la phase de test, nous affectons chaque nouvel objet à la classe dans laquelle il modifie le moins les valeurs des gamma.

Le gamma est défini en fonction des concordances et des discordances. Une autre façon de formuler l'objectif de classement de la phase de test est que nous affections chaque nouvel objet à la classe pour laquelle il modifie le moins le rapport entre les concordances et les discordances.

L'ensemble de N objets est découpé en deux sous-ensembles : l'ensemble d'apprentissage et l'ensemble de test.Les objets O_n sont caractérisés par P descripteurs numériques $D_{n,p}$ avec $n \in [\![1; N]\!]$ et $p \in [\![1; P]\!]$. Les objets appartiennent aux classes C_l avec $l \in [\![1; L]\!]$.

4.6.3.1 La phase d'apprentissage

L'ensemble d'apprentissage utilisé est composé d'objets pour lesquels la vérité terrain (l'appartenance exclusive à l'une des L classes) est connue.

Commençons par construire la matrice

$$\gamma = \begin{pmatrix} \gamma_{1,1} & \cdots & \gamma_{1,P} \\ \vdots & \gamma_{l,p} & \vdots \\ \gamma_{L,1} & \cdots & \gamma_{L,P} \end{pmatrix}$$

où $\gamma_{l,p}$ est le gamma de Goodman et Kruskal du paragraphe 4.6.2.3 entre l'appartenance à la classe C_l et les valeurs du $p^{ème}$ descripteur.

A ce niveau, nous obtenons une matrice indiquant le degré d'aptitude de chacun des P descripteurs à isoler chacune des L classes.

4.6.3.2 La phase de test

Lorsque l'on ajoute un objet O_n dans la classe C_l, on ajoute une quantité de concordances et discordances égale à la quantité de triplets $(n; j; k)$ et $(j; n; k)$ que l'on peut former avec $O_j \in C_l$ et $O_k \notin C_l$.

Donc pour chacun des objets O_n de la base de test, on calcule la matrice

$$\gamma_n = \begin{pmatrix} \gamma^n_{1,1} & \cdots & \gamma^n_{1,P} \\ \vdots & \gamma^n_{l,p} & \vdots \\ \gamma^n_{L,1} & \cdots & \gamma^n_{L,P} \end{pmatrix}$$

où chacun des $\gamma^n_{l,p}$ est le gamma de Goodman et Kruskal obtenu en considérant tous les triplets $(n;j;k)$ et $(j;n;k)$ avec O_n l'objet de l'ensemble de test considéré et (O_j, O_k) tous les couples d'objets de l'ensemble d'apprentissage tels que $O_j \in C_l$ et $O_k \notin C_l$.

Ensuite, nous voulons savoir de quelle classe cet objet O_n a les gamma les plus « proches ». Afin de mesurer ceci, nous calculons la corrélation linéaire de Pearson entre chacune des L séries formées par les lignes de γ_n et la ligne correspondante de γ. On obtient L valeurs comprises entre -1 et 1. En seuillant cette valeur à 0 ou en la normalisant sur $[0, 1]$, nous obtenons une « probabilité » d'appartenance de l'objet O_n à la classe C_l.

Premier exemple : ajout d'un objet à une classe.

La figure 4.15 est un exemple illustrant l'ajout d'un objet à une classe. Il s'agit d'une distribution de 52 objets où les 2 croix rouges sont les 2 objets de la classe C_l et les 50 ronds bleus, les objets des autres classes. La première ligne correspond à l'ensemble d'apprentissage. La deuxième ligne correspond au test d'appartenance à la classe « croix rouge » du nouvel objet représenté par l'étoile. La troisième ligne correspond à l'ensemble obtenu en ajoutant l'étoile à la classe « croix rouge ».

FIGURE 4.15 – Concordances, discordances et gamma correspondants à l'ajout de l'objet O_n (étoile) à la classe C_l (croix rouges).

Commençons par compter les concordances et les discordances sur cette figure. Sur la première ligne, la croix de gauche est plus proche de l'autre croix que de tous les ronds sauf un (celui qui se trouve entre les deux croix). Ce qui donne 49 concordances et 1 discordance. La croix de droite est plus proche de l'autre croix que de tous les ronds sauf deux (les deux les plus à droite). Ce qui donne 48 concordances et 2 discordances. Au total sur la première ligne, nous avons 97 concordances et 3 discordances. Nous pouvons faire les mêmes calculs sur la troisième ligne et nous obtenons 294 concordances pour 6 discordances. Sur la deuxième ligne, l'étoile est plus proche des deux croix que de tous les ronds sauf un. Et les 2 croix sont plus proches de l'étoile que de tous les ronds sauf deux. Ce qui fait un total de 197 concordances et 3 discordances. Le gamma est calculé en fonction des concordances et discordances.

Donc en synthèse de cette exemple :

— La première ligne correspond à la phase d'apprentissage sur cette distribution. $C = 97$, $D = 3$ et $\gamma_{l,p} = 0,94$ sont les concordances, discordances et gamma calculés sur les 100 triplets (i, j, k) avec $i, j \in C_l$ et $k \notin C_l$.

— La deuxième ligne correspond au test de l'objet O_n (l'étoile) sur la classe C_l. $C = 197$, $D = 3$ et $\gamma^n_{l,p} = 0,97$ sont les concordances, discordances et gamma calculés sur les 200 triplets (n, j, k) et (j, n, k) avec $j \in C_l$ et $k \notin C_l$.

— La troisième ligne correspond à la distribution obtenue en ajoutant l'objet O_n à la classe C_l. $C = 294$, $D = 6$ et $\gamma'_{l,p} = 0,96$ sont les concordances, discordances et gamma calculés sur les 300 triplets (i, j, k) avec $i, j \in C_l \cup \{O_n\}$ et $k \notin C_l \cup \{O_n\}$.

Les concordances et les discordances de la troisième ligne sont la somme de celles des deux premières lignes. En conséquence, si le $\gamma^n_{l,p}$ de la phase de test est égal au $\gamma_{l,p}$ de la phase d'apprentissage, ceci signifie que l'ajout de l'objet O_n à la classe C_l ne modifie pas la valeur du $\gamma_{l,p}$.

Deuxième exemple : ajout de différents objets à une classe.

La figure 4.16 présente en première ligne une répartition d'une classe C_l (en rouge) par rapport aux autres classes (en bleu). Le gamma obtenu en phase d'apprentissage vaut $0,67$. Les quatre autres lignes présentent le test de quatre différents objets représentés par l'étoile que l'on essaie d'attribuer à la classe C_l (rouge) et les γ_n obtenus :

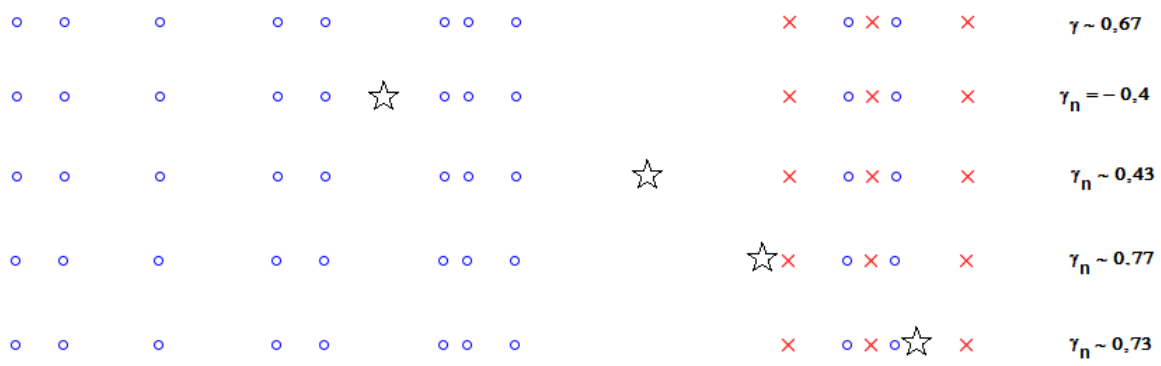


FIGURE 4.16 – Une répartition de la classe C_l (croix rouges) par rapport aux autres classes (ronds bleus) et quatre ajouts d'objets représentés par une étoile.

— Sur la deuxième ligne, l'objet est ajouté dans une zone éloignée des objets de la classe C_l et au milieu d'objets d'autres classes. On obtient $\gamma_n = -0,4$. Cette valeur négative est très différente de la valeur $\gamma \approx 0,67$ de la phase d'apprentissage. L'objet ajouté ne semble pas être de la classe C_l.
— Sur la troisième ligne, l'objet est ajouté dans une zone « vide » située à mi-distance des objets de la classe C_l et de ceux des autres classes. On obtient $\gamma_n \approx 0,43$. Ce qui n'est pas particulièrement proche de la valeur $\gamma \approx 0,67$ de la phase d'apprentissage sans en être non plus particulièrement éloigné. Ce qui semble cohérent avec l'appartenance incertaine de l'objet.
— Sur les quatrième et cinquième lignes, l'objet est ajouté dans une zone proche des objets de la classe C_l. On obtient $\gamma_n = 0,77$ et $0,73$. Ce qui est assez proche de la valeur $\gamma \approx 0,67$ de la phase d'apprentissage. Ce que l'on peut traduire comme une probabilité assez élevée d'appartenance de ces 2 objets à la classe C_l.

À ce stade, nous avons défini notre méthode de classification par corrélation. Il nous reste à l'expérimenter.

4.6.4 Expérimentation sur la classification par genre du challenge MediaEval

La tâche d'étiquetage par genre du challenge MediaEval 2012 [Schmiedeke *et al.*, 2012] est la suite de la tâche de l'année 2011. Il s'agit de reconnaître le genre d'une vidéo parmi 26 genres. La tâche 2012 utilise les données de la tâche d'étiquetage ME12TT.

4.6.4.1 Les données ME12TT

Les objets sont 14 838 épisodes vidéos (shots) tirés de « blip.tv ». 5 288 épisodes ont été placés dans l'ensemble d'apprentissage et 9 550 dans l'ensemble de test. Le partitionnement consiste en l'étiquetage de chaque vidéo par un unique genre parmi 26. Le tableau 4.6 liste les effectifs des 26 genres dans les colonnes 3 à 5. Les effectifs par classe sont très différents, cependant la proportion d'éléments par classe est la même dans l'ensemble d'apprentissage que dans l'ensemble de test.

Id	Genre	Effectif total	Effectif apprentissage	Effectif test	Performance de l'aléatoire
1000	art	505	173	332	4.0 %
1001	autos and vehicles	20	7	13	0.6 %
1002	business	280	96	184	1.9 %
1003	citizen journalism	398	136	262	3.0 %
1004	comedy	510	174	336	3.8 %
1005	conferences and other events	245	84	161	1.6 %
1006	default category	2 240	764	1476	15.3 %
1007	documentary	350	122	228	2.5 %
1008	educational	917	315	602	6.4 %
1009	food and drink	259	87	172	1.8 %
1010	gaming	400	136	264	3.2 %
1011	health	268	92	176	2.0 %
1012	literature	220	82	138	1.6 %
1013	movies and television	864	294	570	6.5 %
1014	music and entertainment	1 124	390	734	8.2 %
1015	personal or auto-biographical	164	57	107	1.2 %
1016	politics	1 098	596	502	5.1 %
1017	religion	862	292	570	6.4 %
1018	school and education	164	58	106	1.2 %
1019	sports	669	228	441	5.0 %
1020	technology	1 307	444	863	9.7 %
1021	the environment	187	64	123	1.4 %
1022	the mainstream media	320	110	210	2.5 %
1023	travel	174	59	115	1.1 %
1024	videoblogging	876	299	577	6.2 %
1025	web development and sites	112	40	72	0.8 %
	TOTAL	14 533	5 199	9 334	3.96 %

TABLE 4.6 – Les effectifs par genre des vidéos de la base ME12TT et la performance de l'aléatoire.

Les descripteurs collectés sur ces vidéos sont de 3 types différents : audio, vidéo et textuel. Le tableau 4.7 liste les différents descripteurs et le nombre de valeurs qui les caractérisent. Le détail de ces descripteurs est donné dans [Mironica *et al.*, 2013].

Nom du descripteur	Dimension	Description
Standard audio descriptors	196	Descripteur audio
MPEG-7	1009	Informations de texture et couleur globales sur toutes les images
Structural descriptors	1430	Informations et relations entre les contours
HoG	81	Le contenu HoG [Ludwig *et al.*, 2009]
Bag-of-Visual-Words rgbSIFT	4096	Les sacs de mots visuels extraits sur les contenus rgbSIFT
TF-IDF of ASR	3466	Les informations textuelles obtenues par reconnaissance de la parole
TF-IDF of metadata	504	Les informations textuelles obtenues des métadonnées (tags, titre...)

TABLE 4.7 – Les descripteurs ME12TT.

4.6.4.2 L'évaluation

L'évaluation des résultats de la tâche de classification par genre de MediaEval est donnée dans l'article [Schmiedeke *et al.*, 2012]. La performance est mesurée comme étant la moyenne des précisions moyennes par classe (genre) :

$$MAP = \frac{1}{L}\sum_{l=1}^{L} AP_l \quad \text{où} \quad AP_l = \frac{1}{N_l}\sum_{k=1}^{N}\frac{f_l(v_k)}{k} \tag{4.18}$$

avec L le nombre de genres, N le nombre de vidéos, N_l le nombre de vidéos de genre l et v_k étant la $k^{ème}$ vidéo de la liste ordonnée $\{v_1, \ldots, v_N\}$. f_l est la fonction qui retourne le nombre de vidéos de genre l parmi les k premières vidéos si v_k est de genre l et 0 sinon.

4.6.5 Performances et résultats

4.6.5.1 Un niveau de performance de référence : le tirage aléatoire

La performance MAP définie au paragraphe 4.6.4.2 prend ses valeurs entre 0% et 100%. Une valeur de 100% signifie que le descripteur a placé pour toutes les classes les objets de la classe systématiquement en tête. C'est le classement optimum. Par contre la valeur nulle n'est pas la performance de l'aléatoire. La valeur nulle reflète un classement plus défavorable que l'aléatoire.

Dans ce contexte, nous avons évalué le niveau de performance de l'aléatoire en appliquant des classements aléatoires aux données de la base ME12TT décrite au paragraphe 4.6.4.1. Nous obtenons expérimentalement une performance de l'aléatoire en moyenne légèrement inférieure à 4%. La dernière colonne du tableau 4.6 liste les 26 performances moyennes par genre AP_l (équation 4.18) et la performance moyenne MAP d'une distribution aléatoire. Il faut bien noter que ces performances sont sensibles aux effectifs. Les classes de plus forts effectifs étant plus probables, elles ont naturellement des performances plus élevées que les autres.

4.6.5.2 Temps de calcul

4.6.5.2.1 Volumétrie

Avec les données de la base ME12TT, pour calculer une colonne de la matrice γ de la phase d'apprentissage décrite au paragraphe 4.6.3.1, il faut examiner plus de 4 milliards de triplets. Ce chiffre est à multiplier par la dimension des descripteurs.

Ainsi, pour un seul des 9 334 shots de la phase de test décrite au paragraphe 4.6.3.2, il faut examiner plus de 50 millions de triplets pour obtenir une colonne de la matrice γ_n. Cette colonne correspond à une seule dimension des descripteurs. Pour l'ensemble des 9 334 vidéos de l'ensemble de test nous obtenons un total de quasiment 500 milliards de triplets. Cette quantité reste encore à multiplier par le nombre de dimension des descripteurs considérés.

Avec les descripteurs audio de dimension 196, nous avons déjà besoin de plusieurs dizaines d'heures pour dérouler la phase d'apprentissage sur un processeur standard. Pour les données vidéos (MPEG-7, Structural descriptors...), les descripteurs sont une centaine de fois plus nombreux. Ce qui multiplie les temps d'autant. Nous avons effectué les calculs en code Scilab (non optimisé) sur un processeur Intel Xeon E5465 2,4 GHz. Le tableau 4.8 recense en colonne « ALL » les temps de calcul de la phase d'apprentissage sur tous les triplets de la base ME12TT pour les 7 descripteurs décrits dans le tableau 4.7. Les chiffres en italique sont obtenus par extrapolation des premières itérations. Les 2 autres colonnes du tableau seront décrites au paragraphe 4.6.5.2.2.

Nous avons donc une méthode qui est particulièrement coûteuse en temps de calcul. Elle n'est pas directement utilisable et nous allons expliquer dans les paragraphes suivants comment résoudre ce problème.

Nom du descripteur	MC10000	MC100000	ALL
Standard audio descriptors	20"	1'50"	2 jours 10 heures
MPEG-7	1'55"	10'00"	7 jours
Structural descriptors	2'40"	12'50"	*12 jours*
HoG	6"	1'00"	0 jours 17 heures
Bag-of-Visual-Words rgbSIFT	12'08"	60'30"	*55 jours*
TF-IDF of ASR	7'10"	43'00"	*14 jours*
TF-IDF of metadata	1'10"	8'40"	2 jours

TABLE 4.8 – Temps de calcul de la phase d'apprentissage.

4.6.5.2.2 Une méthode de Monte-Carlo

Afin de diminuer les temps de calcul, nous introduisons une méthode de Monte-Carlo. C'est à dire que nous ne considérons qu'un échantillon limité de triplets choisis aléatoirement. La figure 4.17 présente l'évolution d'un même $\gamma_{l,p}$ de la phase d'apprentissage en fonction de l'augmentation du nombre de triplets choisis aléatoirement. Le calcul est reproduit 20 fois. Le graphique montre une stabilisation assez rapide des gamma. Au bout de quelques milliers de triplets les gamma ne présentent quasiment plus d'oscillations brusques et semblent se stabiliser. Ceci nous permet d'envisager des méthodes n'utilisant seulement qu'une dizaine de milliers de triplets au lieu de plus de 4 milliards. Nous allons aussi examiner quels sont les gains en temps de calcul et les pertes de précision dues à l'approximation obtenues avec ces méthodes de Monte-Carlo.

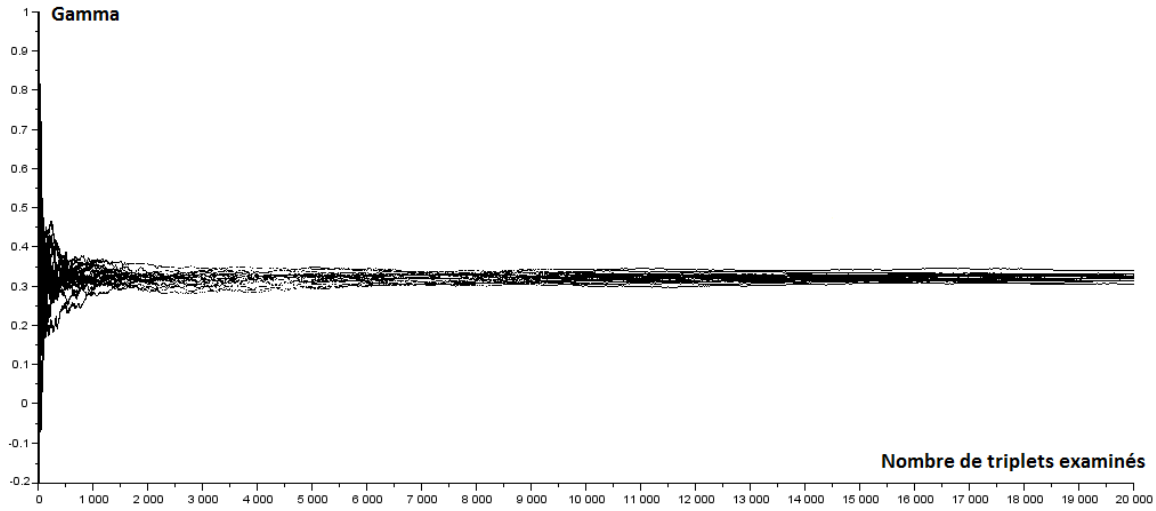


FIGURE 4.17 – 20 calculs du même $\gamma_{l,p}$ de la phase d'apprentissage (même classe, même descripteur) en fonction de l'augmentation du nombre de triplets choisis aléatoirement.

Le tableau 4.9 présente les performances relatives obtenues par notre méthode sur les descripteurs audio standard et vidéos HoG en fonction du nombre de triplets considérés. « MC10000 / MC1000 » signifie que l'on a considéré 10 000 triplets pour la phase d'apprentissage et 1 000 triplets pour la phase de test. Le « ALL » signifie que l'on a considéré la totalité des plus de 4,4 milliards de triplets de la phase d'apprentissage ou la totalité des 50,3 millions de triplets du test d'un shot de la phase de test. Les performances sont relatives au 100% fixé arbitrairement pour la configuration « ALL / ALL ». Nous voyons que les approximations obtenues par ces méthodes de Monte Carlo ne souffrent que de baisses de performances relatives de l'ordre de 0 à 6%. Ces pertes restent à mettre en relation avec le gain en temps de calcul qui est d'un facteur de l'ordre de $\frac{4\,400\,000\,000}{10\,000} = 440\,000$ pour la phase d'apprentissage et de $\frac{50\,300\,000}{1\,000} = 50\,030$ pour la phase de test.

Nom du descripteur	MC10000 / MC1000	MC100000 / MC10000	ALL / MC1000	ALL / MC10000	ALL / ALL
Standard audio descriptors	97%	98%	97.5%	99.5%	100%
HoG	94.5%	96.7%	100%	100%	100%

TABLE 4.9 – Performances relatives selon le nombre de triplets considérés. « MC10000 / MC1000 » signifie que l'apprentissage a été effectué sur 10 000 triplets et le test sur 1 000 triplets. Le « ALL » signifie que l'on a considéré la totalité des triplets.

Les tableaux 4.8 et 4.10 nous donnent les temps des calcul en phase d'apprentissage et de test des codes Scilab sur un processeur Intel Xeon E5465 2,4 GHz. La méthode de Monte Carlo nous permet de diviser les temps de calcul de la phase d'apprentissage d'un facteur 1 000 à 10 000 et la phase de test d'un facteur 100 à 1 000. Ces gains de temps de calcul mis en relation avec la faible perte de performance justifie l'intérêt de cette méthode de Monte Carlo.

Nom du descripteur	MC1000	MC10000	ALL
Standard audio descriptors	2"7	18"	27'45"
MPEG-7	9"	1'30"	1h
Structural descriptors	15"	2'30"	1h20'
HoG	0"6	7"	8'32"
Bag-of-Visual-Words rgbSIFT	35"	6'	10h
TF-IDF of ASR	40"	8'	2h
TF-IDF of metadata	4"	53"	10'

TABLE 4.10 – Temps de calcul du test d'un shot.

Cette méthode devient donc envisageable pour traiter de grandes bases de données.

4.6.5.3 Résultats et comparaisons à d'autres méthodes

[Mironica *et al.*, 2013] utilise 6 méthodes pour la classification par genre sur les mêmes données de la base ME12TT. Les méthodes sont les SVM linéaires, les SVM Radial Basic Function (SVM RBF), les SVM Chi-Square (SVM CHI), les k plus proches voisins (5-NN), les Random Trees (RT) et Extremely Random Forest (ERF) qui ont été présentées au chapitre 3. Le tableau 4.11 présente les performances de ces différentes méthodes. La dernière colonne donne les performances de notre méthode appliquée sans sélection de descripteur mais restreinte à 100 000 triplets pour la phase d'apprentissage et 10 000 triplets pour la phase de test. Ce qui représente respectivement $0,002\%$ et $0,02\%$ de la totalité des triplets.

Nom du descripteur	SVM Linéaire	SVM RBF	SVM CHI	5-NN	RT	ERF	Notre méthode
Std audio descriptors	20.7%	24.5%	35.6%	18.3%	34.4%	42.3%	19.7%
MPEG-7	6.1%	4.3%	17.5%	9.6%	20.9%	26.2%	15.9%
Structural descriptors	7.6%	17.2%	22.8%	8.7%	13.9%	14.9%	8.1%
HoG	9.1%	25.6%	22.4%	17.9%	16.6%	23.4%	8.8%
BOW Visual rgbSIFT	14.6%	17.6%	20.0%	8.6%	14.9%	16.3%	8.5%
TF-IDF of metadata	56.3%	58.1%	48.0%	57.2%	58.7%	57.5%	22.2%

TABLE 4.11 – Performances des différentes méthodes.

Avec la plupart des descripteurs, notre méthode arrive à des performances analogues au SVM linéaire et au 5-NN, mais nettement en dessous des autres classifieurs. Il n'y a que les descripteurs MPEG qui lui permettent de dépasser nettement le SVM linéaire et le 5-NN.

Par ailleurs, si l'on ne considère pas l'efficacité des codes testés, le temps de calcul n'est pas un argument en faveur de notre méthode. Les temps de calcul des autres méthodes dans l'espace de travail Weka (`http://weka.wikispaces.com`) sont tout à fait satisfaisants, du même ordre ou voire meilleurs que ceux de notre méthode. Cependant l'implémentation de notre méthode n'est pas optimisée contrairement aux autres. Il sera intéressant à terme de recoder cette méthode pour réduire ses temps de calcul.

Il est aussi à noter que notre méthode ne donne pas de bons résultats avec les descripteurs

textuels. Ceci n'est pas imputable à la méthode par corrélation de rang en globalité mais uniquement à son aspect « Monte-Carlo ». Les données textuelles contiennent beaucoup de valeurs nulles. Les descripteurs textuels de type métadonnée contiennent en moyenne $98,5\%$ de valeurs nulles. Ceci amène la méthode de Monte Carlo à considérer des triplets qui ne sont pas représentatifs de la globalité. D'où les faibles performances. Pour ces descripteurs textuels, la méthode est applicable sans son aspect Monte-Carlo. Mais dans ce cas, les temps de calcul sont trop importants. Une possibilité de progrès serait peut être d'examiner comment améliorer ces temps en reconnaissant et exploitant algorithmiquement la forte proportion de valeurs nulles de ces descripteurs textuels.

4.6.5.4 Intérêt de la méthode

Cette méthode donne globalement des résultats comparables aux k-NN et aux SVM linéaires mais moins bons que des méthodes comme les SVM non linéaires, les « Random Trees » et les« Extremely Random Forest ».

Cependant le principal intérêt de cette méthode est qu'elle est applicable directement avec des descripteurs non normalisés.

Un autre intérêt de cette méthode est qu'elle ne nécessite pas l'ajustement de paramètres comme il est nécessaire de le faire avec les paramètres du SVM RBF.

De plus, lors de la phase de test chaque classement d'objet est une tâche indépendante des autres classements. Ces tâches peuvent être distribuées sur toutes les ressources matérielles disponibles. La phase de test est donc très facilement parallélisable. La parallélisation et l'adjonction de la méthode de Monte-Carlo permettent d'obtenir une classification d'un grand ensemble d'objets de manière rapide.

4.7 Bilan

Dans la première partie de ce chapitre, nous avons produit une solution personnalisable d'exploration visuelle d'une base de documents et nous l'avons illustrée sur une base vidéo. Nous avons tout d'abord mis au point une méthode de sélection et fusion de descripteurs de types différents (texte et visuels). Le résultat obtenu est une métrique liant toutes les vidéos de la base multimédia. Cette méthode nécessite des données d'entraînement, c'est-à-dire une vérité terrain par paires sur une partie de la base. Cette connaissance par paires est facile à recueillir auprès des utilisateurs car il s'agit simplement de fournir un degré de ressemblance entre documents. Nous avons présenté dans la figure 4.4 un logiciel qui permet de construire cette vérité terrain.

Le principal intérêt de la méthode est d'être automatique en ne nécessitant ni normalisation des données, ni ajustement de paramètres. Nous obtenons une métrique sur la totalité de la base par corrélation de rang entre la vérité terrain partielle et les descripteurs disponibles. Une validation croisée nous a permis de contrôler que la métrique obtenue sur toute la base est une bonne extension de la vérité terrain partielle. Un des intérêts de cette exploration par ressemblance est qu'elle est personnalisable : la ressemblance est construite sur un extrait de la base selon les critères d'appréciation propres à l'utilisateur. Finalement, à l'aide d'une simple méthode de « k plus proches voisins », nous avons obtenu une structuration de la base de la CITIA exploitable par le prototype présenté dans la figure 4.11.

Ensuite, comme nous l'avons vu au chapitre 2, il peut être aussi intéressant d'effectuer

une classification de la base multimédia pour regrouper ensemble les données ressemblantes afin de pouvoir les visualiser. Dans la seconde partie de ce chapitre, nous avons construit une méthode de classification basée sur des corrélations. Cette méthode est elle aussi automatique, ne nécessitant ni normalisation des données, ni ajustement de paramètres. Cependant avec cette méthode de classification nous obtenons des performances qui ne sont pas meilleures que les autres méthodes de l'état de l'art. Nous allons donc explorer dans le chapitre 5 d'autre façon d'effectuer des classifications.

Chapitre 5

Clustering spectral et supervision

Résumé : Dans ce chapitre consacré à l'ajout de supervision dans le Clustering Spectral, nous présentons tout d'abord les méthodes de l'état de l'art du Clustering Spectral semi-supervisé par contraintes. Ensuite nous comparons les résultats de ces méthodes à ceux des techniques de classification supervisée auxquelles nous nous étions déjà comparés au chapitre précédent. Dans une troisième partie, nous nous intéressons au Clustering Spectral semi-supervisé interactif et actif. Nous présentons le procédé de propagation automatique des contraintes par paires pour en proposer une généralisation que nous étudions et validons expérimentalement. Nous terminons ce chapitre par plusieurs études, améliorations et implications de cette propagation sur les méthodes de l'état de l'art du Clustering Spectral semi-supervisé.

Au chapitre 3, dans le paragraphe 3.3 figure un état de l'art sur la classification supervisée ou automatique. Il est en particulier présenté comment le clustering semi-supervisé peut devenir interactif, voir actif. Au paragraphe 3.4, le Clustering Spectral automatique est détaillé. Dans ce nouveau chapitre nous nous intéressons maintenant à la croisée de ces méthodes, soit le Clustering Spectral semi-supervisé, interactif et actif.

Tout d'abord, pour mettre en place une méthode de Clustering Spectral semi-supervisé, il faut savoir de quelle nature est la connaissance et comment l'intégrer dans le processus de semi-supervision. La connaissance peut être absolue et consister en un étiquetage par classe. Recueillir ce type de connaissance n'est pas une tâche simple car le superviseur doit avoir un certain niveau d'expertise, en effet il doit prendre des décisions absolues quand à l'appartenance d'un objet à une classe ou à une autre. Il existe un deuxième type de connaissance souvent utilisée : les contraintes par paires d'objets communément nommées « Must Link » et « Cannot Link ». Ces contraintes indiquent simplement si deux objets sont de la même classe ou non. Elles sont considérées comme étant les plus générales car elles peuvent être extraites d'autres connaissances, comme par exemple d'un étiquetage des objets ou de recommandations d'experts. De plus, ces contraintes correspondent à des annotations par similarité qui sont beaucoup plus faciles à réaliser que des annotations absolues par classe car il est seulement question de savoir si deux objets appartiennent ou non à la même classe. L'expertise nécessaire est donc moins forte.

Le paragraphe 5.1 présente les méthodes de l'état de l'art du Clustering Spectral semi-supervisé par contraintes. Il est en particulier montré comment cette connaissance par paires d'objets peut être intégrée au Clustering Spectral. Le paragraphe 5.2 montre comment adap-

ter le Clustering Spectral semi-supervisé par contraintes pour qu'il puisse gérer une connaissance par étiquetage. Le Clustering Spectral est ensuite mis à l'épreuve sur les données étiquetées du challenge MediaEval [Ionescu *et al.*, 2012] et comparé aux méthodes de l'état de l'art présentées au paragraphe 4.6.

Au paragraphe 5.3, nous examinons comment le Clustering Spectral par contraintes peut devenir interactif et actif. Dans ces processus, il est important d'optimiser les contraintes pour maximiser la qualité de la classification tout en minimisant la sollicitation des experts. La stratégie la plus couramment rencontrée dans des papiers comme celui de Vu, Labroche et Bouchon-Meunier [Vu *et al.*, 2012] consiste en une propagation automatique des contraintes. Cependant, dans la littérature, cette propagation n'est souvent qu'à peine évoquée ou utilisée de façon incomplète. Nous présentons ensuite comment la généraliser. Enfin, après avoir examiné en détail les propriétés de cette propagation, nous la testons sur plusieurs bases de données.

5.1 Clustering Spectral semi-supervisé par contraintes

5.1.1 Des contraintes par paires d'objets

Comme indiqué dans [Xiong *et al.*, 2014], le clustering peut introduire des ambiguïtés sémantiques. Ceci est courant avec la catégorisation d'images car les scènes visuelles sont complexes, les données en entrée sont généralement des descripteurs de bas niveau et le clustering désiré est fortement sémantique. Dans ce contexte, une solution consiste en l'ajout de contraintes par paires fournies par des connaissances externes. Une approche consiste à introduire entre les objets des contraintes « Must Link » (ML) et « Cannot Link » (CL). Les contraintes ML indiquent que les 2 objets sont de la même classe. Les contraintes CL indiquent que les 2 objets ne sont pas de la même classe. Cependant, le choix et le nombre de ces contraintes doivent être optimisés pour respecter la qualité du clustering, tout en gardant un faible coût de calcul et d'annotation. La figure 5.1 présente de telles contraintes ML en vert et CL en rouge sur un ensemble de points répartis en 3 classes.

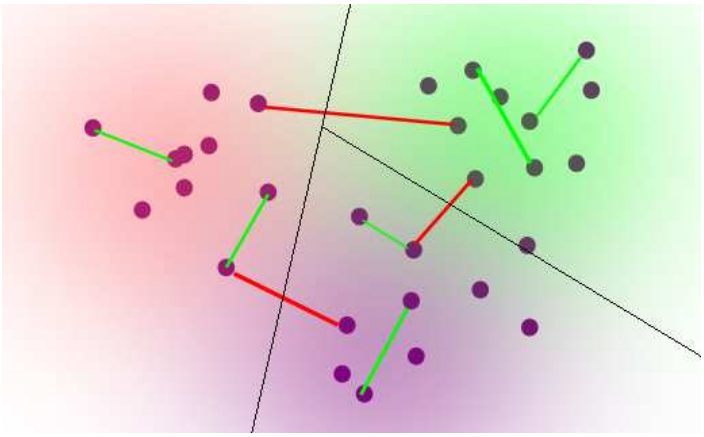


FIGURE 5.1 – Un ensemble de points répartis en 3 classes avec quelques contraintes ML en vert et CL en rouge.

5.1.2 Prise en compte des contraintes

Les contraintes par paires d'objets peuvent être intégrées à chacune des deux premières étapes du Clustering Spectral décrites au paragraphe 3.4.2 :

1. lors de la *construction du graphe de similarité* comme représenté dans la partie gauche de la figure 5.2. C'est le cas de l'Active Clustering (AC) [Xiong *et al.*, 2014], inspiré du Spectral Learning (SL) [Kamvar *et al.*, 2003], où les auteurs proposent d'identifier les objets les plus ambigus, de superviser leurs liens et d'intégrer dans la matrice d'adjacence, W, des 1 pour les ML et des 0 pour les CL. Cependant, il n'y a aucune garantie que les contraintes soient respectées dans le partitionnement final. Néanmoins cette méthode est peu coûteuse en temps de calcul ;

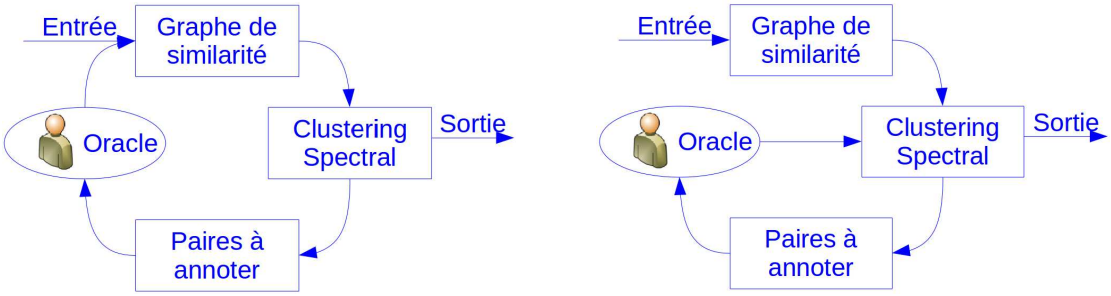


FIGURE 5.2 – Le processus de Clustering Spectral semi-supervisé itératif avec prise en compte des contraintes lors de la construction du graphe de similarité à gauche ou lors de la construction de l'espace spectral à droite.

2. lors de la *construction de l'espace spectral* comme représenté dans la partie droite de la figure 5.2. De nombreuses méthodes existent comme par exemple :
   - « Flexible Constrained Spectral Clustering » [Wang et Davidson, 2010] intègre les contraintes lors de la construction de l'espace spectral et se termine par un k-means ;
   - « Spectral Clustering with Linear Constraints » [Xu *et al.*, 2009] n'utilise pas de k-means et permet seulement un clustering binaire ;
   - « Constrained Clustering via Spectral Regularization » [Li *et al.*, 2009], prend en compte les contraintes dans une étape intermédiaire qui modifie l'espace spectral.

   Une approche différente est proposée avec la méthode « Constrained One-Spectral Clustering » (COSC) [Rangapuram et Hein, 2012]. Les contraintes sont intégrées au calcul grâce à la résolution d'un problème spectral d'optimisation convexe. Le résultat est un bi-partitionnement sans appel à une méthode de partitionnement de type k-means. Cette méthode est étendue au cas du multi-partitionnement grâce à des appels récursifs. Il est montré que COSC respecte très bien les contraintes en offrant un taux d'erreur plus faible que les autres méthodes décrites précédemment. Cependant la méthode COSC est coûteuse en temps de calcul. Elle est en particulier nettement plus coûteuse que l'Active Clustering.

À notre connaissance de la littérature, la prise en compte des contraintes est toujours effectuée comme décrit précédemment, c'est à dire lors des deux premières étapes du Clustering Spectral. Cependant, nous pourrions envisager une introduction des contraintes plus tardive. Elle interviendrait seulement lors du partitionnement des données dans l'espace spectral.

Nous pourrions, par exemple, utiliser les algorithmes MPCK-MEANS [Bilenko *et al.*, 2004] ou COP-KMEANS [Wagstaff *et al.*, 2001] cités au paragraphe 3.3.4 qui permettent à la méthode des k-means de devenir supervisée en intégrant des contraintes ML et CL.

5.2 Clustering Spectral semi-supervisé et étiquetage absolu

Nous avons vu au paragraphe précédent les technique de l'état de l'art du Clustering Spectral semi-supervisé par contraintes. Nous nous intéressons maintenant au cas d'une connaissance dispensée sous forme d'étiquetage par classe. L'objectif est de pouvoir comparer le Clustering Spectral semi-supervisé aux méthodes de classification supervisée présentées dans le chapitre 3.

5.2.1 Approche proposée

On peut facilement transformer ces étiquetages en contraintes par paires d'objets. Il suffit de considérer toutes les paires d'objets étiquetées et si les 2 objets sont de la même classe, nous créons une contrainte ML. Sinon, une CL. De cette façon, nous obtenons un graphe des contraintes complet sur l'ensemble des objets étiquetés. Le Clustering Spectral semi-supervisé par contraintes défini au paragraphe précédent peut ainsi être utilisé avec une connaissance par étiquetage.

En considérant l'ensemble de développement, la connaissance donnée sous forme d'étiquetage est transformée en connaissance sous forme de contrainte ML et CL. Ensuite, ces contraintes sont injectées dans le Clustering Spectral grâce à la méthode COSC. Le graphe de similarité utilisé est un k plus proches voisins avec $k = E(\sqrt{n})$ où n est le nombre d'objets total. La pondération sous forme de similarité gaussienne comme expliqué au paragraphe 3.4.2 est ensuite appliquée.

Dans le cas d'un bi-partionnement, la méthode COSC effectue un découpage en 2 classes en respectant dans la mesure du possible les contraintes ML et CL qui lui sont fournies. C'est-à-dire que lorsque toutes les contraintes sont cohérentes, COSC les respecte. Par contre s'il y a des contraintes incohérentes il ne peut évidemment pas toutes les respecter. La figure 5.3 présente les deux configurations de contraintes avec 3 points qui sont incohérentes dans le cas d'un bi-partitionnement. Les trois contraintes CL de la configuration de gauche indiquent que les 3 points appartiennent à 3 classes différentes. Ce qui est incohérent dans le cas d'un bi-partitionnement. La configuration de droite est incohérente que ce soit en bi comme en multi-partitionnement car les deux contraintes ML indiquent que les 3 points appartiennent à la même classe ; ce qui est contredit par la contrainte CL.

FIGURE 5.3 – Les deux configurations à 3 points incohérentes avec un bi-partitionnement.

Cet éclaircissement sur la notion de contraintes incohérentes étant fait, poursuivons la description de notre approche en examinant la figure 5.4 qui présente un exemple de découpage en 2 classes. Dans la figure 5.4.a, les étoiles représentent les données de l'ensemble de développement et les ronds, celles de l'ensemble de test. Les ellipses indiquent les 4 classes

données par l'étiquetage de l'ensemble de développement. Dans la figure 5.4.b, on voit en vert les contraintes ML obtenues entre les objets de la même classe et en rouge les contraintes CL entre les objets de classes différentes. La figure 5.4.c donne le bi-partitionnement obtenu avec la méthode COSC. Dans ce cas, on remarque que la méthode ne peut pas respecter toutes les contraintes. Les contraintes ML sont tous respectées. Par contre, il est impossible de respecter toutes les contraintes CL.

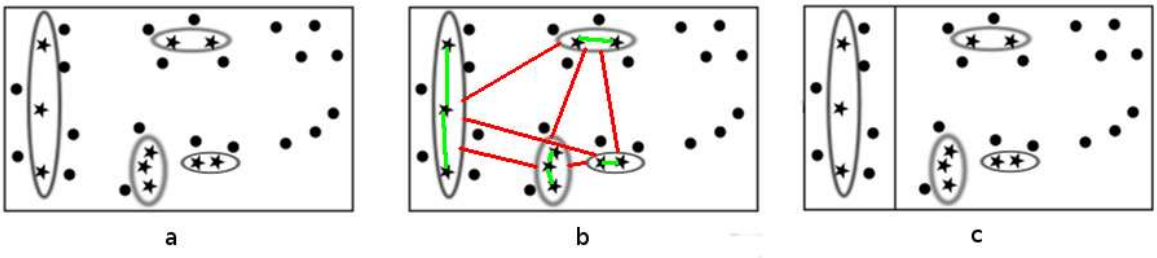


FIGURE 5.4 – Un exemple de clustering COSC en 2 classes.

Si l'on effectue de nouveau une coupe sur l'une des deux classes précédentes, nous obtenons un tri-partitionnement. Nous pouvons répéter l'opération jusqu'à obtenir un partitionnement selon le nombre de classes voulues. Dans le cas d'un multi-partitionnement, la méthode COSC fonctionne de cette façon avec des coupes binaires successives. La figure 5.5 reprend l'exemple précédent avec une coupe en 4 classes. Les étapes intermédiaires sont représentées. Il faut

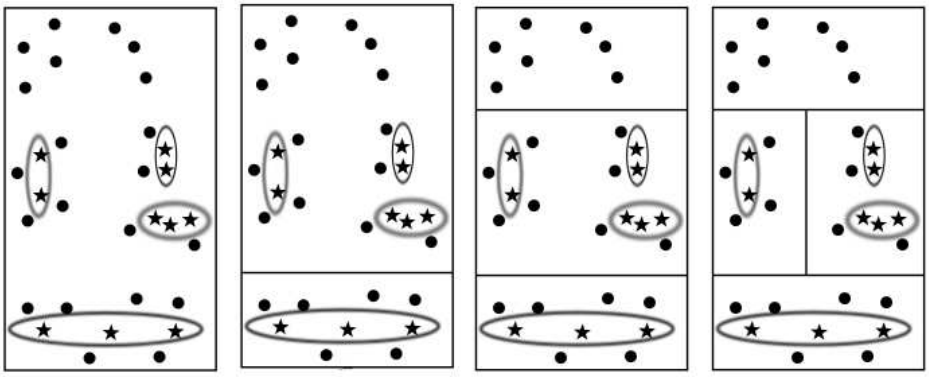


FIGURE 5.5 – Un exemple de clustering COSC en 4 classes.

bien noter qu'avec cette méthode, le partitionnement obtenu respecte toutes les contraintes ML mais pas toutes les contraintes CL. De plus, comme on peut le voir sur la deuxième coupe, lorsque COSC ne peut pas respecter toutes les contraintes CL, l'algorithme de choix peut même décider de n'en respecter aucune. On constate ici une limitation de la méthode : avec l'algorithme COSC, les clusters obtenus peuvent contenir des objets de l'ensemble de développement provenant de plusieurs classes différentes, voir même ne contenir aucun objet de l'ensemble de développement. En conséquence, le clustering peut produire des classes inconnues et fusionner des classes connues.

Nous avons donc décidé de modifier l'algorithme pour qu'il sépare toutes les classes et qu'il ne soit pas alimenté par des contraintes incohérentes. Pour cela, nous avons décidé d'injecter uniquement les contraintes ML et de ne pas fixer le nombre de coupes à effectuer au départ. Nous effectuons des coupes binaires avec l'algorithme COSC **jusqu'à ce que**

toutes les classes de développement soient séparées. C'est ce que nous voyons dans la figure 5.6 avec la coupe supplémentaire finale. Cependant, avec cet algorithme, nous pouvons potentiellement obtenir un plus grand nombre de coupes avec plusieurs clusters ne contenant aucune donnée de développement.

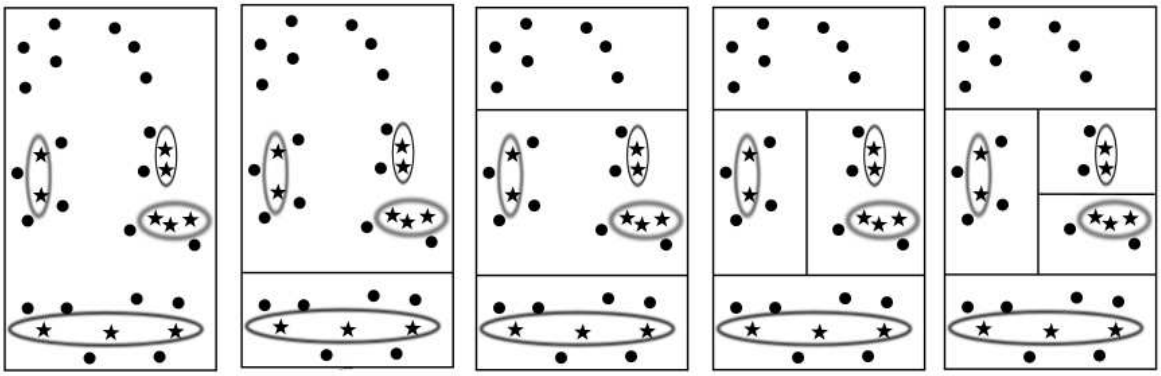


FIGURE 5.6 – Un exemple de clustering COSC modifié pour qu'il sépare toutes les classes de l'ensemble de développement.

5.2.2 Vers un critère d'évaluation : la MAP

Pour pouvoir comparer les performances de la classification obtenues, il faut pouvoir les évaluer en utilisant le critère de performance classiquement employé : la moyenne des précisions moyennes (MAP) définie au paragraphe 4.6.4.2. Pour calculer cette MAP, il faut ordonner les vidéos selon leurs probabilités d'appartenance à chacune des classes. Nous proposons donc de définir une fonction $p_l(v)$ qui mesure une probabilité d'appartenance de la vidéo v à la classe l. Pour toutes les vidéos v de l'ensemble de développement, les annotations nous permettent de spécifier $p_l(v) = 1$ si $v \in l$ et 0 sinon. Pour toutes les vidéos v de l'ensemble de test appartenant au cluster c, on calcule $p_l(v) = \frac{1}{2^{nb(c,l)}}$ où $nb(c,l)$ est le nombre de coupes séparant le cluster c de la classe l. En considérant l'arbre de partitionnement, $nb(c,l)$ est le nombre de liens séparant le cluster c de son plus proche ancêtre commun avec la classe l.

Afin d'illustrer ces propositions, effectuons les calculs correspondants sur l'exemple de la figure 5.7. A gauche, nous avons un partitionnement en 5 clusters $c_1, \cdots, c_5$ et 4 classes $l_1, \cdots, l_4$. A droite, figure l'arbre de partitionnement correspondant aux coupes successives de la figure 5.5. On en déduit la matrice

$$nb = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 3 & 0 & 1 & 1 \\ 4 & 2 & 0 & 1 \\ 4 & 2 & 1 & 0 \\ 2 & 1 & 1 & 1 \end{pmatrix}$$

qui est la matrice des 20 valeurs $nb(c,l)$ entre les 5 clusters et les 4 classes.

De par sa définition, la fonction $p_l(v)$ dépend uniquement de l et de c et toutes les vidéos v de l'ensemble de test qui appartiennent à un même cluster c ont la même probabilité

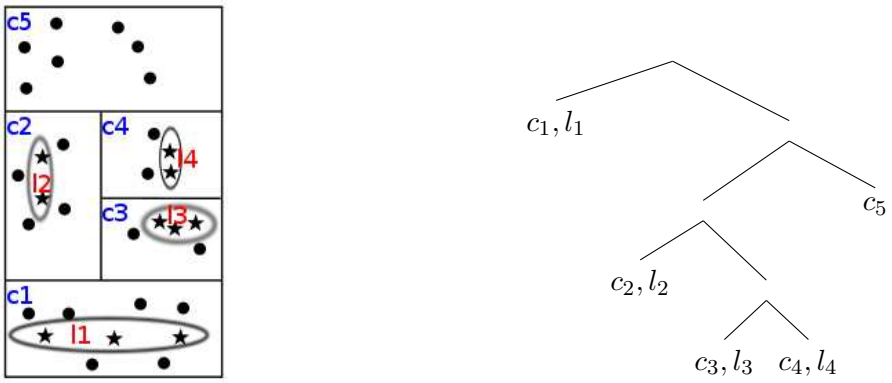


FIGURE 5.7 – Le partitionnement de notre exemple avec son arbre de partitionnement.

d'appartenance à la classe l. On note

$$p = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{8} & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{16} & \frac{1}{4} & 0 & \frac{1}{2} \\ \frac{1}{16} & \frac{1}{4} & \frac{1}{2} & 0 \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{pmatrix}$$

la matrice des 20 valeurs $p_l(c)$ qui représentent les probabilités d'appartenance aux 4 classes $l_1, \cdots, l_4$ pour toutes les vidéos de l'ensemble de développement selon leurs appartenances aux 5 clusters $c_1, \cdots, c_5$.

Nous avons donc les indicateurs nécessaires au calcul d'une MAP classique et nous pouvons alors nous comparer aux méthodes de l'état de l'art.

5.2.3 Application dans le cadre du challenge MediaEval

Nous avons testé cette méthode sur les données du challenge MediaEval composé de 26 classes. Il s'agit des mêmes données que celles utilisées avec la méthode de classification par corrélation au paragraphe 4.6 avec les mêmes ensembles de développement et de test.

Le tableau 5.1 reprend les résultats que nous avons présentés au paragraphe 4.6 : les performances des méthodes de l'état de l'art et celles de notre méthode de classification par corrélation. La dernière colonne présente les performances de notre méthode COSC hiérarchique modifié où nous voyons que les résultats sont proches de notre méthode par corrélation. Pour 4 descripteurs sur 6, nous obtenons de meilleures performances. Cependant les résultats sont toujours de l'ordre de ceux du 5-NN et des SVM linéaires ou RBF. Ils demeurent inférieurs au classifieur le plus efficace dans ce contexte : les Random Forests.

Avec cette méthode, une quantité non négligeable de vidéos se retrouve dans des clusters qui ne contiennent pas de vidéos de l'ensemble de développement. Le tableau 5.2 montre que ce pourcentage d'objets non classés varie entre 2% et 15%. Si nous devons affecter de telles vidéos à une classe, la prise de décision n'est pas simple. Sur notre exemple, $p_{l_2}(c_5) = p_{l_3}(c_5) = p_{l_4}(c_5)$ implique qu'il n'est pas possible de décider si les objets du cluster c_5 appartiennent à la classe l_2, l_3 ou l_4. Il apparaît intéressant d'adapter la méthode pour forcer le classement de ces objets.

Nom du descripteur	SVM Lin.	SVM RBF	SVM CHI	5-NN	RT	ERF	MC	COSC modifié
Std audio desc.	20.7%	24.5%	35.6%	18.3%	34.4%	42.3%	19.7%	25.2%
MPEG-7	6.1%	4.3%	17.5%	9.6%	20.9%	26.2%	15.9%	12.5%
Structural desc.	7.6%	17.2%	22.8%	8.7%	13.9%	14.9%	8.1%	8.0%
HoG	9.1%	25.6%	22.4%	17.9%	16.6%	23.4%	8.8%	12.1%
BOW Visu SIFT	14.6%	17.6%	20.0%	8.6%	14.9%	16.3%	8.5%	9.7%
TF-IDF	56.3%	58.1%	48.0%	57.2%	58.7%	57.5%	22.2%	40.6%

TABLE 5.1 – Performances des méthodes de l'état de l'art comparées aux nôtres.

Nom du descripteur	Pourcentage de vidéos non classées
Standard audio descriptors	8.9%
MPEG-7	14.3%
Structural descriptors	6.4%
HoG	8.8%
BOW Visu SIFT	2.1%
TF-IDF of metadata	11.4%

TABLE 5.2 – Pourcentage de vidéos non classées avec la méthode de COSC hiérarchique modifié.

5.2.4 Pistes d'amélioration

La figure 5.8 présente une adaptation de l'algorithme hiérarchique précédent qui force le classement de tous les objets grâce à une prise en compte des contraintes CL de manière progressive. Dans cet algorithme, nous effectuons toujours des coupes binaires successives par l'algorithme COSC en injectant au départ uniquement les contraintes ML. L'algorithme est illustré sur l'exemple de la figure 5.8 en 8 étapes successives :

1. On commence avec le même exemple de 31 points répartis en 4 classes en considérant uniquement les contraintes ML.
2. La première coupe binaire est effectuée.
3. Ensuite, lorsqu'une coupe crée un cluster sans échantillon de l'ensemble de développement, la coupe est mise en suspens et représentée en « pointillé ». On isole ces données indépendantes et on poursuit le clustering sur les autres données.
4. Ce principe est appliqué jusqu'à ce que l'on obtienne une coupe qui sépare les données de développement dans deux clusters différents.
5. Dès qu'une telle coupe est trouvée, les contraintes CL en sont déduites et toutes les coupes en « pointillé » sont annulées.
6. La coupe suivante respecte toutes les contraintes ML et CL. L'algorithme est ensuite poursuivi depuis l'étape 3 de la même façon jusqu'à ce que tous les clusters contiennent exactement une classe.
7. Une optimisation est effectuée : lorsqu'un cluster contient exactement 2 classes, les contraintes CL sont injectées.
8. Cette étape n'est qu'une optimisation qui permet d'obtenir directement la coupe finale en évitant la création de coupes en « pointillé » inutiles dans ce cas.

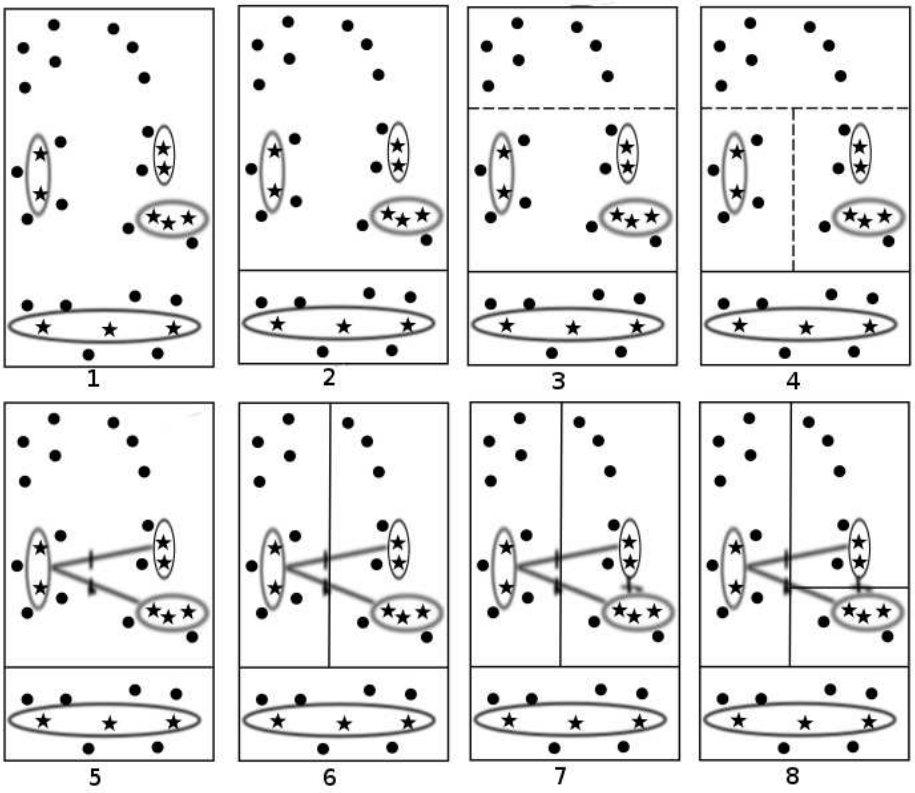


FIGURE 5.8 – Un nouvel algorithme COSC hiérarchique supervisé.

Une telle méthode améliore forcément les performances. Cependant les objets non classés ne sont pas majoritaires et les améliorations pressenties ne semblent pas suffisantes pour atteindre les meilleures performances de l'état de l'art. La mise en œuvre de cette méthode, faute de temps, demeure une perspective de travail à venir.

Dans la suite, nous nous sommes intéressés à un autre aspect de l'introduction de supervision qui semble plus prometteur. Il s'agit du Clustering Spectral semi-supervisé avec l'introduction progressive de la connaissance dans un processus interactif complété par une propagation automatique des contraintes.

5.3 Clustering Spectral semi-supervisé interactif

Nous avons vu au chapitre 3 qu'un clustering semi-supervisé peut devenir interactif dès lors qu'il s'inscrit dans une démarche itérative qui enchaîne de manière répétitive un clustering et une supervision par un expert. Nous avons aussi vu qu'en rajoutant une étape de sélection des paires à annoter à soumettre à l'expert, nous obtenons un clustering semi-supervisé actif.

Les schémas présentés dans les figure 5.2 sont déjà des schémas de Clustering Spectral semi-supervisés actifs. Dans la suite, nous nous intéressons à l'amélioration de ces méthodes par un processus d'augmentation automatique du nombre des contraintes par paires : la propagation automatique des contraintes.

5.3.1 Propagation automatique des contraintes et généralisation

Comme nous l'avons vu au début du 5.2, des contraintes par paires peuvent être incohérentes. Il est nécessaire de les éviter pour maintenir un fonctionnement correct du Clustering Spectral. Or, les experts humains étant perfectibles, après un ajout de contraintes, des ambiguïtés voisines peuvent être résolues automatiquement grâce à des méthodes de propagation sans appel supplémentaire à l'annotation experte. De plus, cette propagation est très intéressante pour réduire le coût d'acquisition de données expertes en évitant de solliciter inutilement l'expert. Elle permet enfin d'atteindre une meilleure qualité de partitionnement avec un coût de calcul inférieur car elle évite des itérations inutiles.

L'état de l'art propose déjà des méthodes que nous généralisons dans cette section :

- Règle 1 : $ML + ML \Rightarrow ML$:
  Dans la figure 5.9, nous voyons que si un objet est relié à deux autres par des ML alors il est de la même classe que les deux autres. Donc les 3 objets sont de la même classe. Nous obtenons bien un troisième ML. Cette règle est nommée transitivité de la relation ML.
- Règle 2 : $ML + CL \Rightarrow CL$:
  Avec la figure 5.10, nous voyons deux objets reliés par un ML. Ils sont donc de la même classe. Si l'un de ces deux objets est relié à un troisième par un CL alors il n'est pas de la même classe. Nous avons donc deux points appartenant à la même classe et un troisième appartenant à une autre classe. Nous avons donc entre ces points un ML et deux CL. Cette règle est nommée combinaison des relations ML et CL.

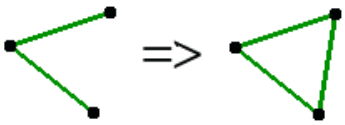


FIGURE 5.9 – $ML + ML \Rightarrow ML$ (Règle 1)

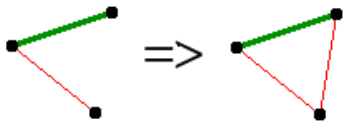


FIGURE 5.10 – $ML + CL \Rightarrow CL$ (Règle 2)

- Règle 3 : $CL + CL \Rightarrow ?$:
  Cette configuration illustrée dans la figure 5.11 donne une indétermination dans le cas général. Cependant, tel que présenté dans [Mallapragada *et al.*, 2008], le cas du bi-partitionnement résout cette indétermination avec $CL + CL \Rightarrow ML$ (voir la figure 5.12). Effectivement dans le cas d'un partitionnement en uniquement 2 classes C_1 et C_2, si un objet X appartient à la première classe C_1 et si X n'est pas dans la même classe que 2 autres objets Y et Z. Alors forcément Y et Z appartiennent à la deuxième et même classe C_2.

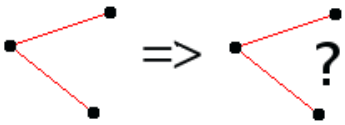


FIGURE 5.11 – en multi-partitionnement : $CL + CL \Rightarrow ?$

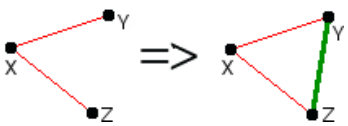


FIGURE 5.12 – en bi-partitionnement : $CL + CL \Rightarrow ML$

Ce cas n'est pas anecdotique. Dans la littérature, il existe beaucoup de bi-partitionnement. Par exemple, dans [Rangapuram et Hein, 2012], la méthode COSC est évaluée dans 3

des 5 cas par des bi-partitionnements. Cependant, dans ces évaluations, aucune propagation n'est considérée.

Notre première contribution est de proposer une généralisation de la troisième configuration de propagation des contraintes qui exploite la combinaison des deux contraintes CL (règle 3).

À notre connaissance, dans le cas d'un partitionnement en 3 classes ou plus, la configuration $CL+CL$ est toujours mentionnée comme étant indéterminée. Cependant, nous pouvons quand même déduire quelque chose d'une configuration ne comportant que des CL. Comme présenté dans la figure 5.13, dans un tétraèdre en tri-partitionnement, si nous avons 5 arêtes CL alors la sixième et dernière arête est forcément un ML. Effectivement, en prenant 3 classes C_1, C_2 et C_3 et la configuration de la figure 5.13 comme W et X sont liés par un CL alors forcément W et X appartiennent à 2 classes différentes C_1 et C_2. Comme Y et Z sont eux mêmes reliés à W et X par des CL alors ils ne font pas partie des classes C_1 et C_2 et forcément ils appartiennent à la même classe restante C_3. Par conséquent, ils sont reliés par un ML.

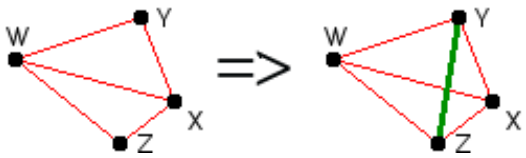


FIGURE 5.13 – En tri-partitionnement, dans le tétaèdre : $5 \times CL \Rightarrow ML$.

Le cas du tétraèdre en tri-partitionnement se généralise pour tout entier n au n-simplexe dans le cas d'un n-partitionnement. Dans un n-simplexe, si l'on a $\left(\frac{n(n-1)}{2} - 1\right)$ arêtes CL alors la dernière et $\frac{n(n-1)}{2}$ème arête est forcément un ML. La démonstration est itérative de manière analogue au cas du tétraèdre en tri-partitionnement. La figure 5.14 présente le cas du n-simplexe pour n allant de 4 à 7.

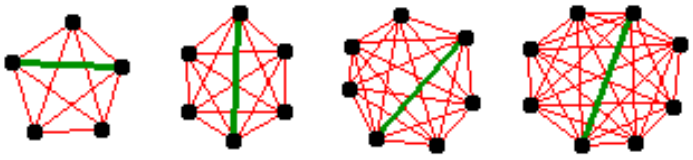


FIGURE 5.14 – En n-partitionnement, dans le n-simplexe : $\left(\frac{n(n-1)}{2} - 1\right) \times CL \Rightarrow ML$.

Nous présentons au paragraphe 5.3.4 les résultats expérimentaux qui mettent en évidence l'impact de ces propagations dans le processus itératif du clustering spectral semi-supervisé interactif. Mais avant de passer aux expérimentations, nous pouvons déjà mettre en évidence deux bénéfices différents de la propagation des contraintes. C'est ce qui est présenté dans le paragraphe suivant.

5.3.2 Bénéfices de la propagation des contraintes

La propagation automatique des contraintes agit à 2 niveaux dans les processus de supervision :

1. Lors de la sélection des contraintes, elle évite de retenir des contraintes qu'il est inutile de soumettre à l'expert car on peut les obtenir automatiquement. On évite ainsi des ambiguïtés potentiellement apportées par un expert.
2. Lors de la prise en compte des contraintes, elle augmente le nombre de contraintes à injecter dans la méthode de classification semi-supervisée. On peut espérer une convergence plus rapide en moins d'itérations.

Il est acquis que la première action améliore tous les processus de supervision quels qu'ils soient. La figure 5.15 illustre ce phénomène. Dans la partie haute de la figure, on voit un graphe de n points. Il a au maximum $n(n-1)/2$ liens. Donc sans propagation, il est nécessaire de solliciter un expert $n(n-1)/2$ fois pour couvrir l'ensemble des contraintes. Dans le cas d'un bi-partitionnement, nous voyons dans la partie basse de la figure que la propagation automatique des contraintes (en traits continus) crée des contraintes propagées (en pointillé). Elle permet de réduire le nombre de paires à superviser de $n(n-1)/2$ à $n-1$. Dans le cas, d'un partitionnement multi-classes, il y a aussi une diminution du nombre de paires à superviser. L'ampleur de cette diminution dépend de la configuration du nuage de points et de l'ordre selon lequel les paires sont supervisées.

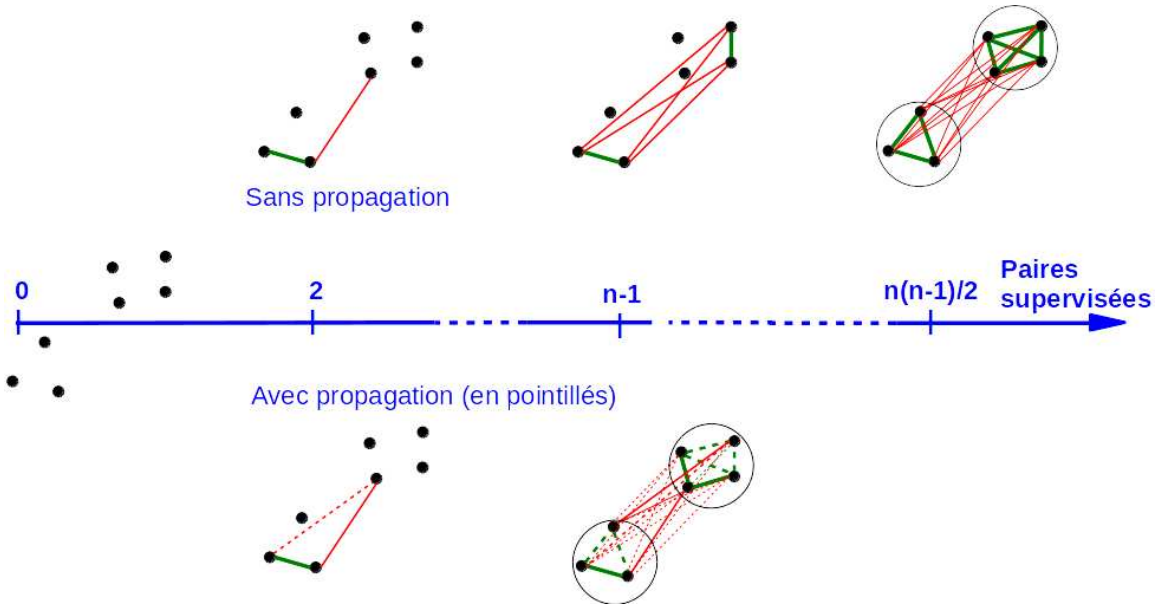


FIGURE 5.15 – En bi-partitionnement, un nuage de n points comporte $n(n-1)/2$ paires à superviser. Avec l'utilisation de la propagation, il y a au maximum $n-1$ paires à superviser.

Nous pouvons aussi noter que cette première action bénéfique de la propagation a une conséquence intéressante en lien avec la discussion sur la distinction entre expert et Oracle effectuée au chapitre 3. Dans le cas où l'on effectue une propagation complète après chaque paire supervisée l'expert peut être qualifié d'Oracle. Effectivement, si l'on ne soumet pas à l'expert de paires qui peuvent être obtenues automatiquement par propagation, il ne peut pas créer de contradiction.

En ce qui concerne la seconde action, selon les méthodes et leur prise en compte des contraintes, l'influence des contraintes propagées dans le clustering peut avoir plus ou moins d'effets.

1. Une méthode qui respecte les contraintes sans aucune violation est forcément insensible

à cette deuxième action. La figure 5.16 illustre ce phénomène. A gauche, si le partition-

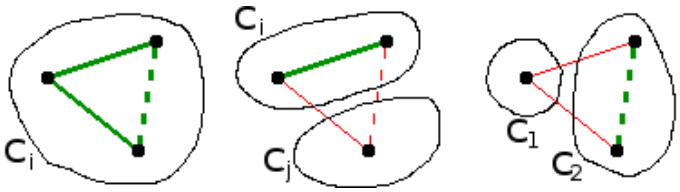


FIGURE 5.16 – Un partitionnement qui respecte les contraintes connues (arêtes continues) respecte forcément les contraintes déduites (en pointillé).

nement respecte les 2 contraintes ML (arêtes vertes continues), alors les 3 sommets sont placés dans la même classe. Et donc forcément, la contrainte ML déduite (arête verte en pointillé) est respectée. Au centre, si le partitionnement respecte les 2 contraintes ML et CL (arêtes verte et rouge continues), alors les 2 premiers sommets sont placés dans une classe différente de celle du troisième sommet. Et forcément, la contrainte CL déduite (arête rouge en pointillé) est respectée. A droite, pour la troisième règle de propagation en bi-partitionnement, si les 2 contraintes CL sont respectées, alors le premier sommet appartient à la première classe, tandis que les 2 autres appartiennent à la deuxième classe. Et donc la contrainte ML déduite est respectée. Le même phénomène de respect des contraintes déduites existe avec la généralisation de la troisième règle au cas du n-partitionnement.

2. Au contraire, une méthode ne respectant pas les contraintes peut gagner à utiliser la propagation. Par exemple, l'« Active Clustering » présenté au paragraphe 5.1.2 n'est pas une méthode qui cible le respect des contraintes. Elles sont injectées dans le graphe de similitude afin de guider la coupe effectuée par le Clustering Spectral. Beaucoup de contraintes ne sont pas respectées. En augmentant le nombre d'arêtes concernées par ce pré-conditionnement, la propagation automatique des contraintes améliore forcément la qualité de cette méthode. C'est ce phénomène qui est illustré en 3 étapes dans la figure 5.17. L'étape initiale, à gauche, présente un graphe avec 2 contraintes ML en vert et la coupe obtenue en bleue. En haut figure sa matrice de similarité contrainte avec les $+1$ des ML. L'étape 2, au centre diffère juste de la supervision d'un lien additionnel : une contrainte ML en rouge. En bleu, sans propagation, on voit que la coupe est inchangée ; la supervision d'un lien n'est pas suffisante à elle seule pour modifier la coupe du graphe. Par contre, à l'étape 3, on voit que la propagation augmente le nombre de contraintes CL et provoque une modification de la coupe du graphe. C'est cet effet d'accélération de la convergence vers la coupe finale que nous nommons le deuxième bénéfice de la propagation.

Comme nous l'avons déjà expliqué au paragraphe 5.2, dans le cas des bi-partitionnements et la plupart des autres cas, la méthode COSC [Rangapuram et Hein, 2012] respecte les contraintes qui sont cohérentes entre elles. Elle est donc insensible au deuxième bénéfice de la propagation. Avec COSC, le principal effet obtenu par la propagation des contraintes est la non sollicitation inutile de l'expert. Par contre l'Active Clustering est sensible aux deux bénéfices de la propagation. On pressent déjà que la propagation va apporter plus d'améliorations à la convergence de l'Active Clustering qu'à celle de COSC. On peut donc se poser la question qui est de savoir si la propagation peut permettre à un clustering plus simple et moins gourmand en ressources de calcul de rivaliser avec COSC.

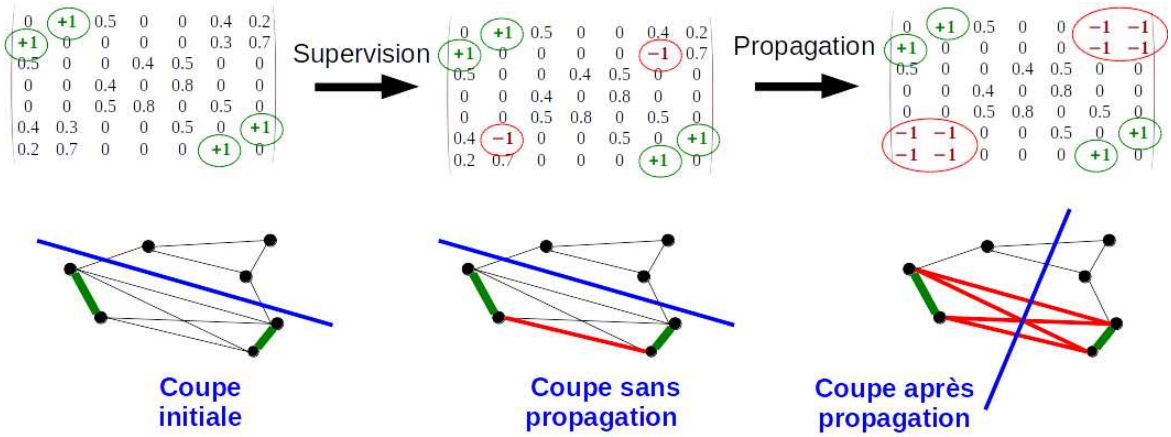


FIGURE 5.17 – Deuxième bénéfice de la propagation automatique des contraintes.

À ce point, nous avons généralisé la propagation et identifié ses bénéfices. Nous pouvons examiner comment implémenter cette propagation et nous pouvons mettre en place un processus complet intégrant la propagation automatique des contraintes pour tirer profit de ces bénéfices. C'est ce que nous présentons au paragraphe suivant.

5.3.3 Implémentation de la propagation

La propagation automatique des contraintes est un processus qui peut devenir coûteux en temps de calcul et consommation mémoire. Il est important de bien connaître les propriétés de la propagation pour mettre en œuvre une implémentation efficace. Dans cette section, nous commençons par examiner les propriétés itératives des 3 règles de propagation dans le but d'identifier un enchaînement efficace des règles. Ensuite, dans le processus complet d'ajout de contraintes propagées nous montrons où appliquer ces règles pour réduire l'espace de recherche. Pour finir nous examinons les différentes possibilités de stockage des données et d'implémentations, pour les améliorer et retenir la solution la plus adaptée.

5.3.3.1 Propriétés itératives de la propagation

Pour obtenir une propagation totale, la première règle de propagation doit être appliquée plusieurs fois. La figure 5.18 illustre ceci avec un exemple à 2 itérations. Au départ, il y a deux ML existants (en traits continus) et un ML ajouté (en pointillé). A la fin de la première itération, il y a deux nouveaux ML propagés (en pointillé). Et il faut une deuxième itération pour obtenir le dernier ML propagé et ainsi une propagation complète. Ceci n'est qu'un exemple car le nombre d'itérations nécessaires pour atteindre la convergence de la règle 1 peut être plus grand.

Il faut noter que lorsque la convergence de la règle 1 est obtenue, les composantes connexes du graphe « Must Link » sont totalement connectées. La conséquence de ceci est qu'ensuite la règle 2 ne nécessite qu'une seule itération. Ce phénomène est illustré par l'exemple de la figure 5.19. Au départ, nous retrouvons en vert les six ML initiaux et propagés avec en pointillé un CL ajouté. Ensuite, en une seule itération, tous les CL sont propagés car les quatre points avec leurs six ML forment une composante connexe totalement connectée.

La troisième règle de propagation examine uniquement des liens CL pour ne créer que

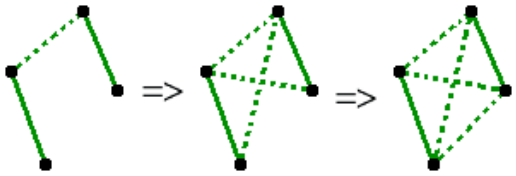


FIGURE 5.18 – Propagation de la règle 1 : $ML + ML \Rightarrow ML$.

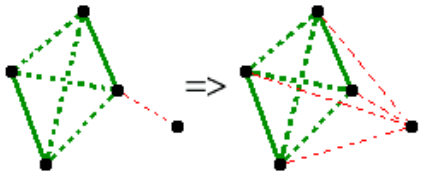


FIGURE 5.19 – Propagation de la règle 2 : $ML + CL \Rightarrow CL$

des liens ML. Elle ne crée pas de CL donc cette règle n'est pas itérative. Cependant si des liens ML sont créés par cette troisième règle, il convient de recommencer la propagation des 3 règles depuis le début, jusqu'à obtenir la propagation totale.

Nous avons ainsi un procédé de propagation complet qui enchaîne les 3 règles. Nous pouvons donc ajouter cette étape de propagation entre les étapes de supervision et de clustering pour obtenir le processus complet représenté dans la figure 5.20.

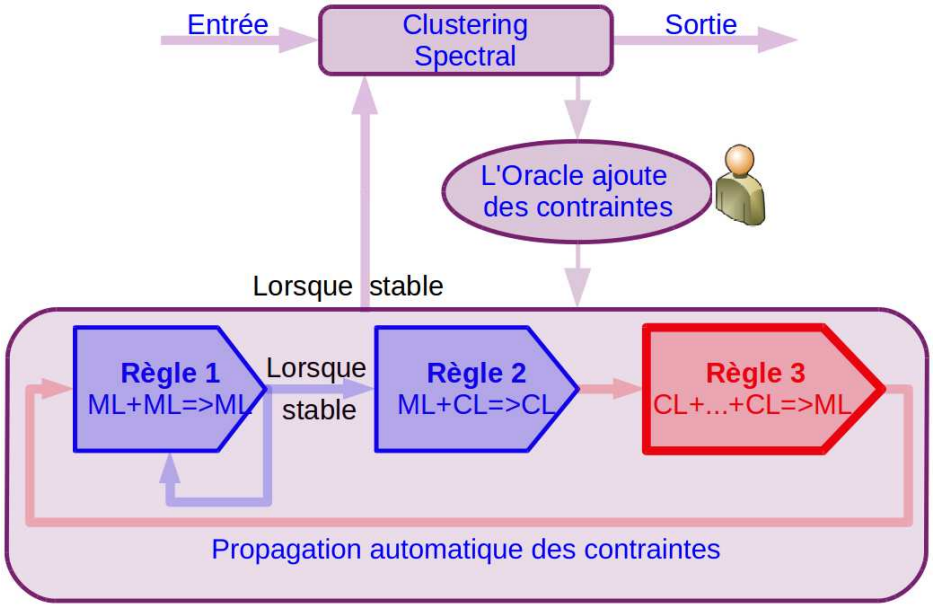


FIGURE 5.20 – Le processus complet de Clustering Spectral semi-supervisé interactif avec propagation totale des contraintes.

5.3.3.2 Optimisation : restriction aux nouveaux liens ajoutés

Nous venons de voir les propriétés itératives de la propagation automatique et comment les étapes d'ajout de nouvelles contraintes et de propagation s'enchaînent. ML et CL sont les ensembles des contraintes « Must Link » et « Cannot Link ». ML_{add} et CL_{add} sont les sous-ensembles des contraintes ajoutées. Elles sont représentées en pointillé dans les figures 5.18 et 5.19. On voit qu'il n'est pas utile d'examiner les combinaisons de tous les liens entre eux. On peut restreindre l'examen aux combinaisons impliquant au moins un lien qui vient d'être ajouté. On peut donc effectuer les optimisations suivantes :

- la règle 1 peut être restreinte à chaque itération aux combinaisons $ML_{add} + ML$,

- la règle 2 peut être restreinte aux combinaisons $ML_{add} + CL$ et $ML + CL_{add}$,
- la règle 3 peut être restreinte à un examen de tous les n-simplexes composés d'au moins un CL_{add}.

Comme tout au long du processus, le nombre de contraintes augmente, ces restrictions présentent l'intérêt de ne pas réexaminer les propagations déjà effectuées et donc d'optimiser fortement le code à implémenter.

5.3.3.3 Optimisation : stockage des contraintes et implémentation des deux premières règles

Il existe plusieurs méthodes pour stocker les contraintes et implémenter ces règles de propagation. Elles sont présentées ici dans un ordre d'efficacité croissante :

1. La première méthode est basée sur des listes ml, cl. Les propagations sont effectuées par des parcours de listes avec des boucles imbriquées. La propagation selon la règle 1 est donnée par l'algorithme 2. L'algorithme de la règle 2 est analogue sans la boucle « repeat » car cette règle n'est pas itérative.

Algorithm 2 - Propagation automatique selon la règle 1 : $ML + ML \Rightarrow ML$

Require: ml
  repeat
    $ml_{added} \leftarrow \emptyset$
    for all $a \in ml$ **do**
      for all $b \in ml$ avec $b \neq a$ **do**
        if (a et b ont une extrémité commune) et ($a + b \notin ml \cup ml_{added}$) **then**
          Ajouter $a + b$ à ml_{added}
        end if
      end for
    end for
    $ml \leftarrow ml \cup ml_{added}$
  until $ml_{added} \neq \emptyset$
  return ml

2. La deuxième méthode est une amélioration de la méthode précédente grâce à la restriction aux nouveaux liens ajoutés comme décrit dans la section précédente. Les listes ml_{add} et cl_{add} contiennent les liens ajoutés. La propagation selon la règle 1 est alors donnée par l'algorithme 3. L'algorithme de la règle 2 est toujours analogue sans la boucle « repeat ».
3. La troisième méthode est basée sur un stockage des contraintes et l'utilisation de calculs matriciels. L'intérêt de cette méthode réside dans la compacité du code et dans l'utilisation de bibliothèques de calcul matriciel optimisées.

   La propagation selon la règle 1 est alors donnée par l'algorithme 4. L'algorithme de la règle 2 est encore analogue sans la boucle « repeat ».
   ML est la matrice d'adjacence des liens « Must Link ».
   La matrice ML^2 donne les nombres de chemins « Must Link » de longueur 2.
   $I(M)$ est la matrice indicatrice définie par $I(M) = \begin{pmatrix} 1 \; si \; M_{i,j} \neq 0 \\ 0 \; si \; M_{i,j} = 0 \end{pmatrix}$.

Algorithm 3 - Propagation automatique selon la règle 1 : $ML_{add} + ML \Rightarrow ML$

Require: ml, ml_{add}
  $ml_{lastadded} \leftarrow ml_{add}$
  repeat
    $ml_{added} \leftarrow \emptyset$
    for all $a \in ml_{lastadded}$ **do**
      for all $b \in ml$ avec $b \neq a$ **do**
        if (a et b ont une extrémité commune) et ($a + b \notin ml \cup ml_{added}$) **then**
          Ajouter $a + b$ à ml_{added}
        end if
      end for
    end for
    $ml_{lastadded} \leftarrow ml_{added}$
    $ml_{add} \leftarrow ml_{add} \cup ml_{added}$
    $ml \leftarrow ml \cup ml_{added}$
  until $ml_{added} \neq \emptyset$
  return ml, ml_{add}

Algorithm 4 - Propagation automatique selon la règle 1 : $ML + ML \Rightarrow ML$

Require: ML
  repeat
    $ML_{12} \leftarrow ML + ML^2$ {Matrice des chemins de longueur 1 et 2}
    $ML_{added} \leftarrow I\,(ML_{12} - Diag(ML_{12})) - ML$
    $ML \leftarrow ML + ML_{added}$
  until $ML_{added} \neq 0$
  return ML

Exemple : avec le graphe de la figure 5.21, on a $ML = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$, $ML^2 = \begin{pmatrix} 2 & 1 & 1 & 1 \\ 1 & 2 & 1 & 1 \\ 1 & 1 & 3 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}$, $ML_{added} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$ et pour finir on obtient le graphe de la figure 5.22 avec $ML_{(new)} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}$

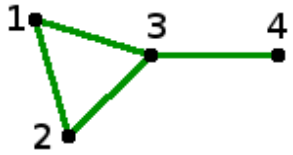


FIGURE 5.21 – Un exemple de graphe simple avant la propagation.

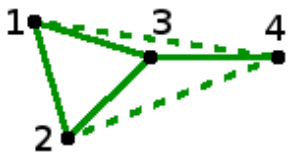


FIGURE 5.22 – Propagation du graphe simple donnée dans la figure 5.21.

4. La quatrième méthode est une amélioration de la méthode précédente grâce à la restriction aux nouveaux liens ajoutés comme décrit dans la section précédente. ML_{add} et CL_{add} sont les matrices (sparses) qui contiennent les liens ajoutés. La propagation selon la règle 1 est alors donnée par l'algorithme 5. L'algorithme de la règle 2 reste analogue sans la boucle « repeat ».

Algorithm 5 - Propagation automatique selon la règle 1 : $ML_{add} + ML \Rightarrow ML$

Require: ML, ML_{add}
  $ML_{added} \leftarrow ML_{add}$
  repeat
    $ML_{12} \leftarrow ML + ML \times ML_{added} + (ML \times ML_{added})^T$
    $ML_{added} \leftarrow I\left(ML_{12} - Diag\left(ML_{12}\right)\right) - ML$
    $ML_{add} \leftarrow ML_{add} + ML_{added}$
    $ML \leftarrow ML + ML_{added}$
  until $ML_{added} \neq 0$
  return ML, ML_{add}

Dans un environnement de développement pourvu de bibliothèques de calcul matriciel efficaces comme la librairie Linear Algebra Package (LAPACK) [Anderson *et al.*, 1990], l'algorithme 5 doit être privilégié. Sinon, l'algorithme 3 est tout à fait utilisable.

5.3.3.4 Stockage des contraintes et implémentation de la troisième règle

Pour ce qui concerne la troisième règle de propagation, dans le cas d'un bi-partitionnement, elle est de la forme $CL+CL \Rightarrow ML$ et peut être traduite en opération matricielle. On obtient un algorithme analogue à l'algorithme 5 sans la boucle « repeat ».

Dans le cas d'un partitionnement en n classes avec $n > 2$, la règle 3 consiste en un examen de tous les n-simplexes composés de CL et impliquant un CL ajouté. Ceci peut être implémenté par des algorithmes récursifs. Comme tout algorithme récursif, il peut aussi s'écrire de manière itérative avec des boucles imbriquées. Quoi qu'il en soit, la règle 3 se présente comme étant coûteuse lorsque n devient grand.

À ce point, nous pouvons implémenter le processus complet intégrant la propagation automatique des contraintes pour pouvoir expérimenter et valider l'intérêt de ces propagations.

5.3.4 Les résultats expérimentaux

Les validations expérimentales sont effectuées sur deux types d'ensembles de données : des données synthétiques avec différents niveaux graduels de séparation des classes allant de très séparé à totalement mélangé et des données réelles Blip10000 [Schmiedeke *et al.*, 2013] issues de la classification par genre de vidéos. Les deux ensembles de données sont utilisés dans des configurations de partitionnement bi, tri comme multi-classes. Nous avons expérimenté les deux méthodes de Clustering Spectral mentionnées à la section 5.1.2 :

- la méthode Active Clustering (AC) [Xiong *et al.*, 2014], qui ajoute les contraintes à la matrice du graphe d'adjacence et qui ne garantit pas le respect des contraintes ;
- Constraint One Spectral Clustering (COSC) [Rangapuram et Hein, 2012], qui intègre les contraintes dans la phase de construction de l'espace propre et cible le respect des contraintes.

Pour évaluer les performances, nous utilisons l'indice de Rand normalisé présenté au chapitre 3 qui consiste en une normalisation du ratio du nombre de paires classées de la même façon dans les deux partitions sur le nombre de paires totales. Ses principaux avantages sont de prendre ses valeurs dans l'intervalle $[-1, 1]$ où la valeur 1 signifie que les deux partitions sont identiques et où surtout la valeur 0 signifie que les deux partitions sont indépendantes. Nous calculons l'indice de Rand entre le partitionnement de la vérité terrain et le partitionnement obtenu.

Nous avons adopté la procédure de validation suivante [Xiong *et al.*, 2014] : la méthode de clustering est appliquée une première fois sans contrainte. Puis à chaque itération, de nouvelles contraintes sont intégrées au processus. Elles sont alors propagées automatiquement et la méthode de clustering est appliquée de nouveau. Pour être indépendant et ne pas privilégier un algorithme de clustering sur un autre, les contraintes sont choisies aléatoirement parmi toutes les paires possibles. Nous nous assurons que l'ensemble des propagations sont réalisées grâce à notre méthode (figure 5.20). La performance des résultats est ensuite comparée à la vérité terrain grâce à l'indice Rand normalisé. Dans nos expérimentations, les contraintes sont extraites automatiquement de la vérité terrain qui consiste en un étiquetage de tous les objets. Dans le cadre d'une utilisation réelle, les contraintes proviendraient directement de l'appel à un expert qui devrait dire si 2 objets sont liés ou non.

5.3.4.1 Le cas bi-classes synthétique

Dans ces expérimentations, aucune normalisation n'est appliquée. Le graphe des similarités est construit à l'aide d'un 5 plus proches voisins symétrique. La pondération gaussienne est utilisée.

Nous avons généré 5 ensembles de données synthétiques de 100 points répartis en 2 classes (voir la partie droite de la figure 5.23) avec les propriétés suivantes :

- *1er ensemble de données* : a ses points placés aléatoirement ;
- *2ème ensemble de données* : est partiellement mélangé avec deux classes circulaires qui se chevauchent partiellement ;
- *3ème ensemble de données* : a deux classes disjointes mais contigües ;
- *4ème ensemble de données* : similaire au troisième mais avec une zone de séparation entre les deux classes ;
- *5ème ensemble de données* : a deux classes fortement séparées.

Les 5 graphiques de la figure 5.23 correspondent aux résultats sur les 5 ensembles de données décrits précédemment. Ils présentent les évolutions de la qualité en fonction du nombre de paires sélectionnées aléatoirement. Les courbes en noir correspondent à une sélection/supervision de paires aléatoires, sans aucune propagation. En bleu, le processus est complété par une propagation automatique suivant les 2 premières règles. En rouge, la troisième règle de propagation est ajoutée. Chaque courbe correspond aux valeurs moyennes de 20 exécutions différentes. Les courbes en pointillé correspondent à la méthode COSC, les courbes en trait continu, à l'Active Clustering. On constate tout d'abord que l'Active Clustering est fortement amélioré par la propagation automatique. L'amélioration apportée à la méthode COSC est sensible lorsque l'on traite des configurations complexes (les deux premiers ensembles de données), mais imperceptible dans les autres cas.

Dans tous les cas, avec l'utilisation des 3 règles de propagation, nous avons la garantie d'atteindre le partitionnement réel en moins de 100 paires supervisées,

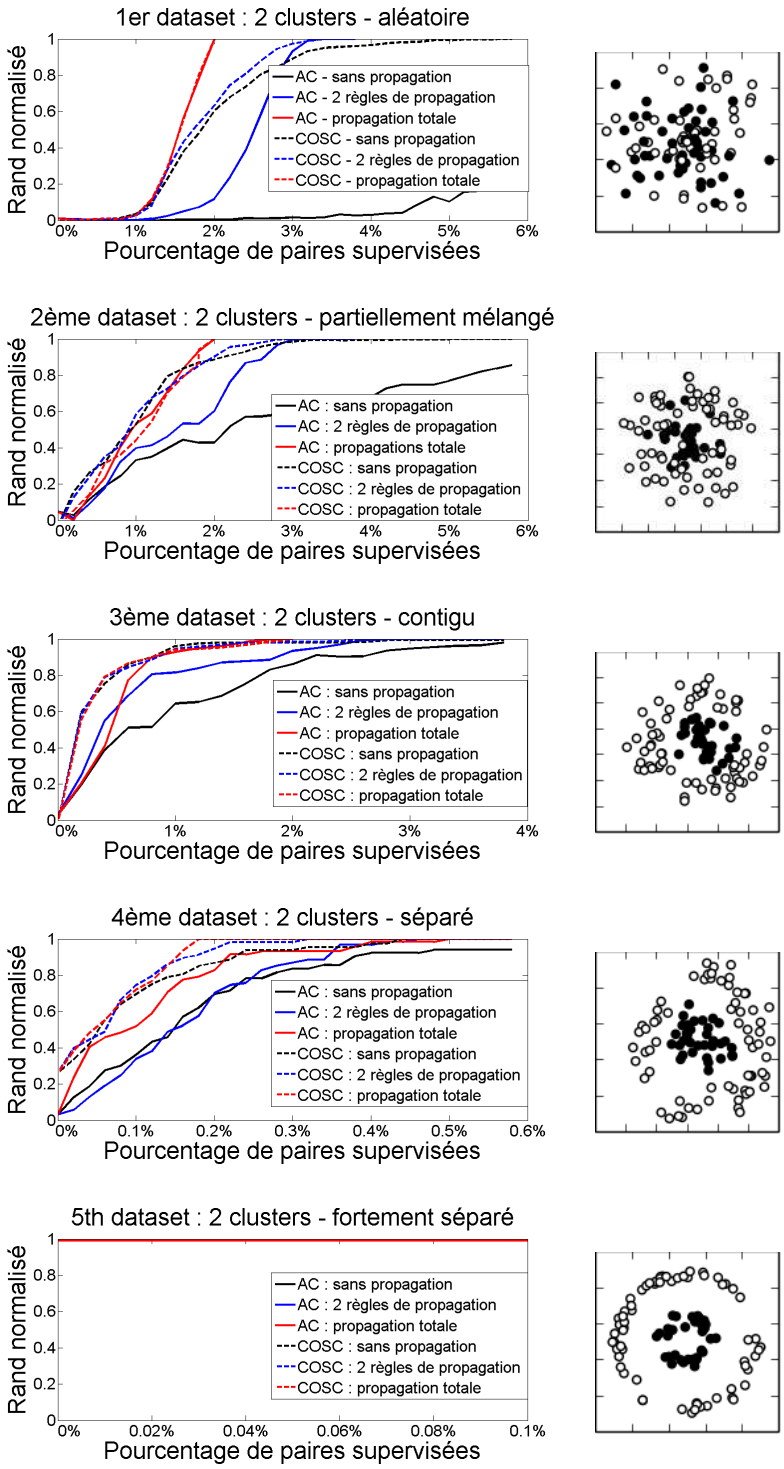


FIGURE 5.23 – Qualité du partitionnement en fonction du nombre de paires supervisées avec l'Active Clustering (traits continus) et COSC (pointillé) en utilisant aucune propagation (en noir), les 2 premières (en bleu) ou les 3 règles de propagation (en rouge).

c'est à dire en supervisant moins de 2% de toutes les paires possibles. Dans les configurations complexes (les deux premiers ensembles de données), la propagation permet à COSC de converger 2 fois plus rapidement que sans propagation. L'Active Clustering converge moins vite que COSC mais le facteur d'amélioration apporté par la propagation automatique est plus grand. Dans un cas simple comme le cinquième ensemble de données, les 2 méthodes obtiennent le partitionnement réel sans avoir besoin de supervision.

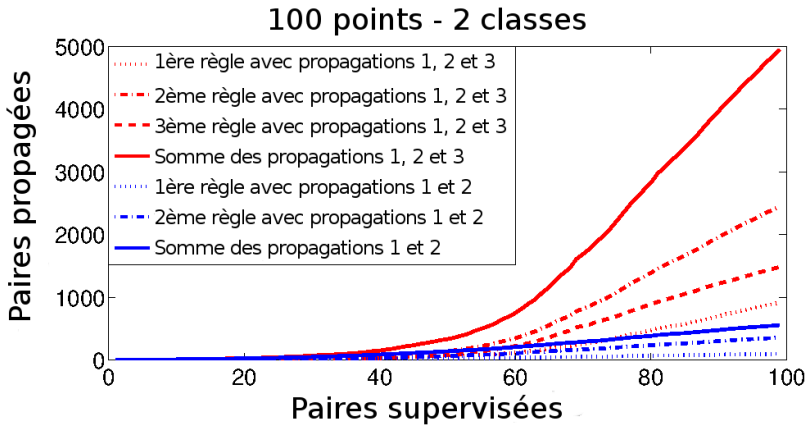


FIGURE 5.24 – Nombre de paires propagées selon l'usage ou non de la troisième règle dans le cas d'un **bi-partitionnement**.

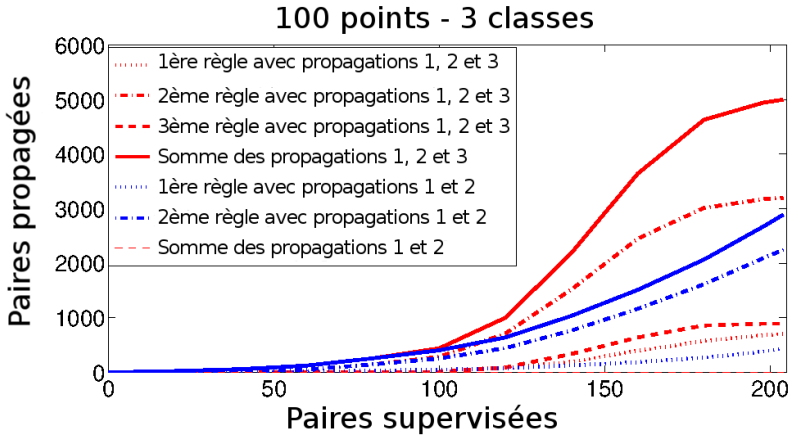


FIGURE 5.25 – Nombre de paires propagées selon l'usage ou non de la troisième règle dans le cas d'un **tri-partitionnement**.

Les figures 5.24 et 5.25 présentent le nombre de paires propagées en fonction du nombre de paires choisies aléatoirement et supervisées. La différence entre les deux figures est le nombre de classes. La figure 5.24 est un bi-partitionnement. La figure 5.25 est un tri-partitionnement.

Avec le bi-partitionnement de la figure 5.24, une lecture graphique nous indique qu'après 80 paires supervisées, il y a :

- 2800 paires propagées par l'utilisation conjointe des règles 1, 2 et 3 (courbe rouge en trait continu),
- 850 paires propagées uniquement par la règle 3 dans le cas de l'utilisation conjointe des

règles 1, 2 et 3 (la courbe rouge en tirets) et

- 400 paires propagées par l'utilisation conjointe des règles 1 et 2 (courbe bleue en trait continu).

2800 est une quantité bien plus importante que la somme de 400 et 850. On constate bien qu'il y a un effet en « cascade » des propagations en utilisant conjointement les 3 règles de propagation.

Les constations précédentes sont extensibles du bi vers le multi-partitionnement. Effectivement dans le cas du tri-partitionnement de la figure 5.25, nous obtenons les mêmes constations que précédemment. Après 150 paires supervisées, il y a :

- 3000 paires propagées par l'utilisation conjointe des règles 1, 2 et 3 (courbe rouge en trait continu),
- 500 paires propagées uniquement par la règle 3 dans le cas de l'utilisation conjointe des règles 1, 2 et 3 (la courbe rouge en tirets) et
- 1300 paires propagées par l'utilisation conjointe des règles 1 et 2 (courbe bleue en trait continu).

La règle 3 est certes coûteuse et apporte relativement peu de contraintes à elle seule mais elle permet une propagation significativement plus importante lorsqu'elle est combinée avec les deux autres règles.

Obtention d'un effet de seuil

Avec les 2 premiers ensembles de données de la figure 5.23, on remarque que la qualité de la partition commence à augmenter seulement lorsque l'on a déjà ajouté une quantité importante de contraintes. Cet effet de seuil est dû à la progression non linéaire du nombre de contraintes apportées par la propagation automatique. Les figures 5.24 et 5.25 montrent tout d'abord qu'en dessous d'un certain seuil le nombre de paires propagées est peu important. L'explication de ce phénomène provient du fait que les paires sont choisies aléatoirement et que donc, au début, les objets ne sont que peu connectés. Ensuite, lorsque le graphe des contraintes commencent à être plus connexe, les propagations deviennent importantes. On constate aussi que la contribution de la règle 3 est non négligeable dans les cas des bi et tri-partitionnements. Son usage conjoint avec les 2 autres règles permet un gain significatif de contraintes.

5.3.4.2 Le cas bi-classes réel

Nous avons reproduit la même expérimentation qu'au paragraphe 5.3.4.1 avec deux ensembles de données réels de 100 séquences vidéos réelles de l'ensemble de données Blip1000 [Schmiedeke *et al.*, 2013]. Les données exploitées sont des vecteurs *descripteurs audio standards* de dimension 196 proposés dans [Mironica *et al.*, 2013]. Pour les deux ensembles de données, nous avons retenus 50 vidéos de deux genres : « Santé » et « Littérature » pour le premier ; « Santé » et « Documentaire » pour le deuxième.

Les résultats obtenus sont présentés dans la figure 5.26. Ils sont analogues à ceux obtenus avec les ensembles de données synthétiques de référence et pour la méthode COSC vus dans la section précédente. On note l'intérêt de l'ajout de la règle 3 dans le cas de « Santé » et « Documentaire ». En terme de vitesse de convergence, « Santé » et « Littérature » nous donnent des comportements intermédiaires entre ceux des troisième et quatrième ensembles de données synthétiques. Ce qui semblerait indiquer que les vidéos de genre « Santé » et

« Littérature » ont des données audios assez séparées. « Santé » et « Documentaire » nous donnent des résultats analogues à ceux du deuxième ensemble de données synthétiques. Ce qui semblerait indiquer que les vidéos de genre « Santé » et « Documentaire » ont des données audio partiellement mélangées pour le descripteur utilisé.

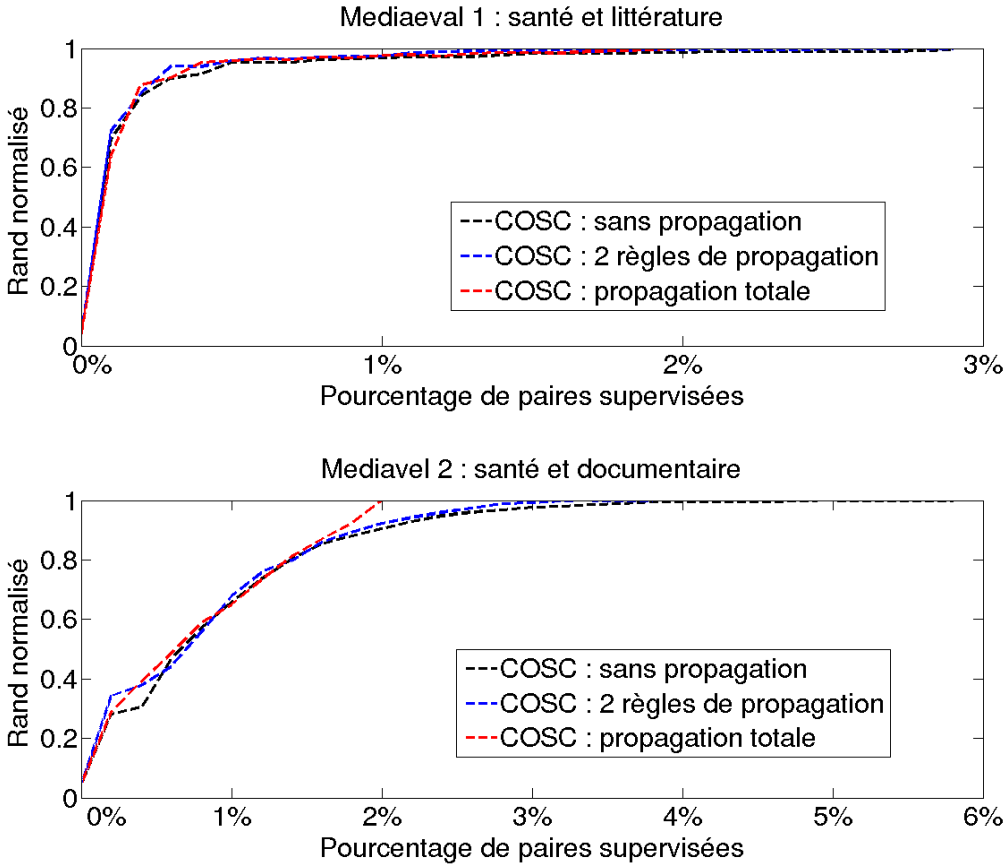


FIGURE 5.26 – Qualité du partitionnement en fonction du nombre de paires supervisées avec COSC en utilisant aucune propagation (en noir), les 2 premières (en bleu) ou les 3 règles de propagation (en rouge).

5.3.4.3 Le cas multi-classes

Nous avons reproduit la même expérimentation qu'au paragraphe 5.3.4.1 sur des ensembles de données multi-classes. Le premier ensemble de données est composé de 100 points placés aléatoirement sur le disque unité bidimensionnel et répartis en 3 classes équilibrées. Il est représenté dans la partie droite de la figure 5.27. Les deuxième et troisième ensembles de données sont composés des données vidéos réelles décrites au paragraphe 5.3.4.2. Le deuxième ensemble de données est composé de 100 vidéos des genres « Santé », « Documentaire » et « Littérature » réparties en 3 classes égales. Le troisième ensemble de données est composé des 5197 vidéos réparties en 26 classes inégales. Il s'agit de l'ensemble des données du challenge MediaEval pour l'année 2012.

Les résultats sont présentés dans la partie gauche de la figure 5.27. La courbe bleue correspond à l'utilisation des 2 premières règles de propagation. La courbe rouge correspond

à l'ajout de la troisième règle que nous avons proposé au paragraphe 5.3.1.

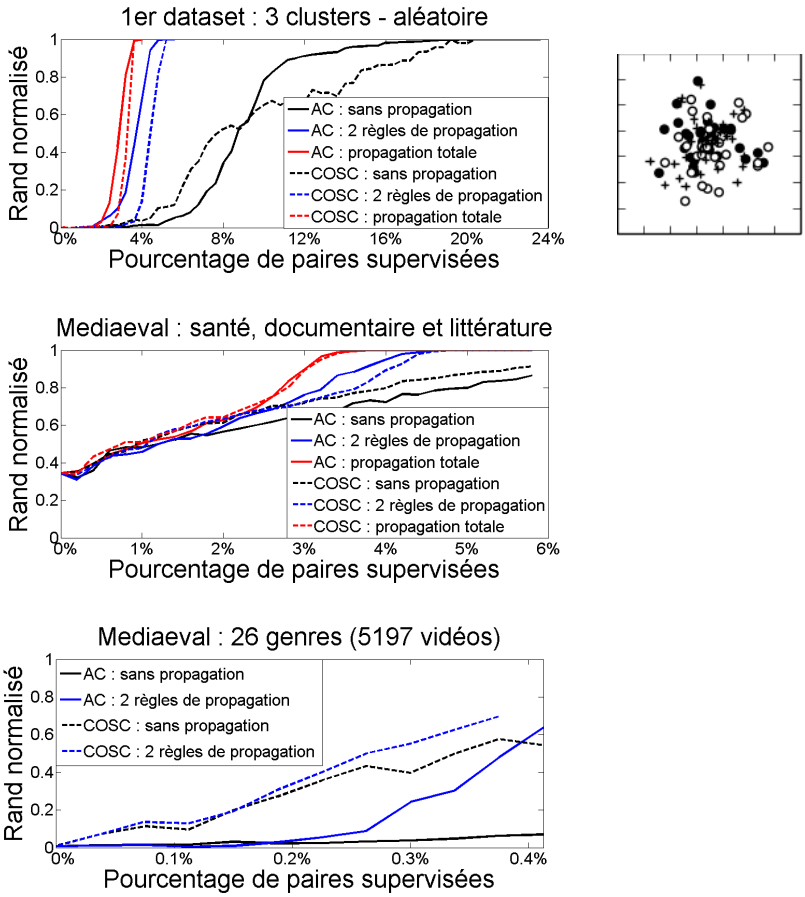


FIGURE 5.27 – Qualité du partitionnement en fonction du nombre de paires supervisées avec l'Active Clustering (traits continus) et COSC (pointillé) en utilisant aucune propagation (en noir), les 2 premières (en bleu) ou les 3 règles de propagation (en rouge).

On peut observer que dans ce nouveau cas d'utilisation, la propagation automatique apporte un gain de vitesse de convergence significatif. Plus précisément, les deux premiers jeux de données montrent l'avantage de la troisième règle de propagation proposée dans la section 5.3.1 qui permet d'obtenir le partitionnement parfait avec 20% de contraintes en moins que la propagation limitée aux deux premières règles. Dans cette expérimentation, COSC atteint même une performance inférieure à l'Active Clustering. Ceci peut s'expliquer par le fait que COSC effectue des coupes binaires de manière hiérarchique, ce qui n'est pas adapté à des ensembles de données tri-classes.

Avec le troisième ensemble de données qui contient un plus grand nombre de données et clusters, COSC surpasse l'Active Clustering. Une fois encore, les méthodes sont bien toutes deux améliorées par la propagation automatique des contraintes. Les résultats montrent que les deux premières règles de propagation, appliquées à 50 000 liens, ce qui représente seulement $0,37\%$ de tous les liens possibles, permettent d'améliorer la qualité de partitionnement de 21% pour COSC et de 650% pour l'Active Clustering.

Cependant, dans un tel cas, il faut discuter des coûts de calcul de la technique de propagation. En effet, les deux premières règles peuvent être appliquées efficacement grâce à la

vectorisation de produit de matrices sparses. Mais la troisième règle consiste en un examen de tous les 26-simplexes du graphe. Pour l'instant, une telle analyse n'est pas optimisée et devient rapidement trop coûteuse en temps de calcul et en consommation mémoire. En conséquence, seules les deux premières règles de propagation ont été appliquées. D'autres travaux pourront être conduits pour améliorer ces aspects et supporter le passage à l'échelle.

Nous avons vu au paragraphe 5.3.4.1 qu'avec des paires choisies totalement aléatoirement, il existe un effet de seuil dans le processus de propagation. Il est lié au fait qu'au début les contraintes ne sont que peu connectées entre elles et qu'il y a donc que très peu de propagations. Nous allons examiner maintenant comment réduire cet effet de seuil pour obtenir dès le début des itérations du processus, un nombre de propagations conséquent.

5.3.5 Améliorations du processus de Clustering Spectral actif avec propagation des contraintes

5.3.5.1 Amplification de l'effet de la propagation grâce à une stratégie de sélection des contraintes

Depuis le début, les paires d'objets ont toujours été sélectionnées totalement aléatoirement. Cependant, nous avons vu à la fin du paragraphe 3.3.4, que des stratégies de sélection active existent et peuvent améliorer la convergence de ces méthodes actives. Nous proposons ici une sélection qui va amplifier les effets de la propagation automatique des contraintes. Une façon simple et efficace est de restreindre la sélection aux paires aléatoirement choisies à celles composées d'un objet **relié** à d'autres objets par des liens déjà supervisés et d'un objet **non encore relié** à d'autres objets par des liens déjà supervisés.

La figure 5.28 compare cette stratégie de sélection « aléatoire reliée » à la sélection « totalement aléatoire » sur le même ensemble bi-classes que celui utilisé dans la figure 5.24.

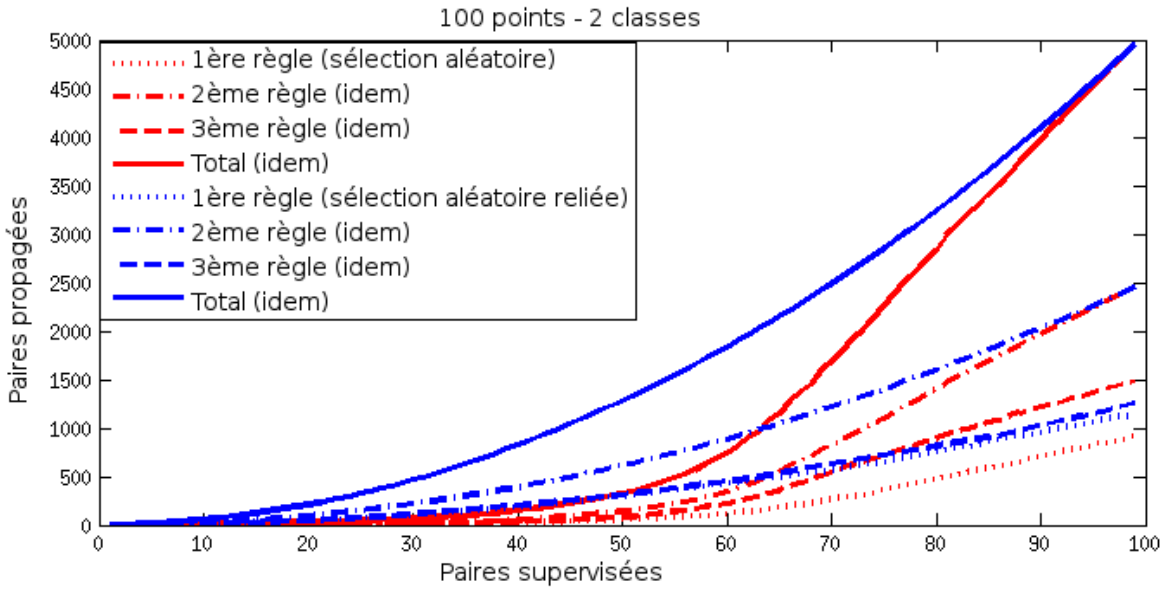


FIGURE 5.28 – Nombre de contraintes propagées en fonction du nombre de paires supervisées avec un ensemble de 100 points équirépartis en 2 classes aléatoires. Comparaison de la sélection de paires aléatoire avec la stratégie de sélection aléatoire reliée.

Nous observons que les courbes « aléatoire reliée » sont nettement au dessus des courbes « totalement aléatoire », et ceci dès le début du processus. Ce procédé de sélection a donc bien pour effet d'amplifier la propagation des contraintes dès le début du processus. Une lecture graphique nous montre qu'après seulement 20 paires supervisées, la sélection aléatoire propage 30 paires, tandis que la stratégie de sélection « aléatoire reliée » en propage 210. Ce qui représente une amplification de facteur 7. Cette stratégie est intéressante car elle peut permettre au processus d'annotation de s'arrêter plus tôt si un objectif de qualité de partition partielle est donnée. Bien sûr, à la fin, les deux sélections se rejoignent en convergeant vers une annotation totale.

5.3.5.2 Orchestration des tâches d'annotation, propagation et clustering

Le processus complet de Clustering Spectral actif avec propagation automatique généralisée des contraintes donné dans la figure 5.20 est séquentiel. Les trois étapes que sont le clustering, la supervision et la propagation sont effectuées les unes après les autres. Et de plus, la propagation automatique des contraintes peut devenir coûteuse en temps de calcul dès lors que le nombre de points et le nombre de classes deviennent importants. Ceci peut engendrer des temps d'inaction de l'opérateur de supervision qui peuvent devenir un problème.

Dans le but de diminuer le temps d'attente de l'expert entre deux supervisions successives, notre processus peut être rendu parallèle. Comme illustrée dans la figure 5.29, la propagation automatique des contraintes peut être effectuée en parallèle des étapes de clustering et supervision. Il faut cependant rajouter une étape de fusion des contraintes issues de la propagation avec celles de la supervision. Cette étape n'est pas coûteuse. Selon l'option de stockage des contraintes du paragraphe 5.3.3 choisie, il s'agit uniquement de la fusion de deux listes ou d'une somme ou opération logique sur deux matrices.

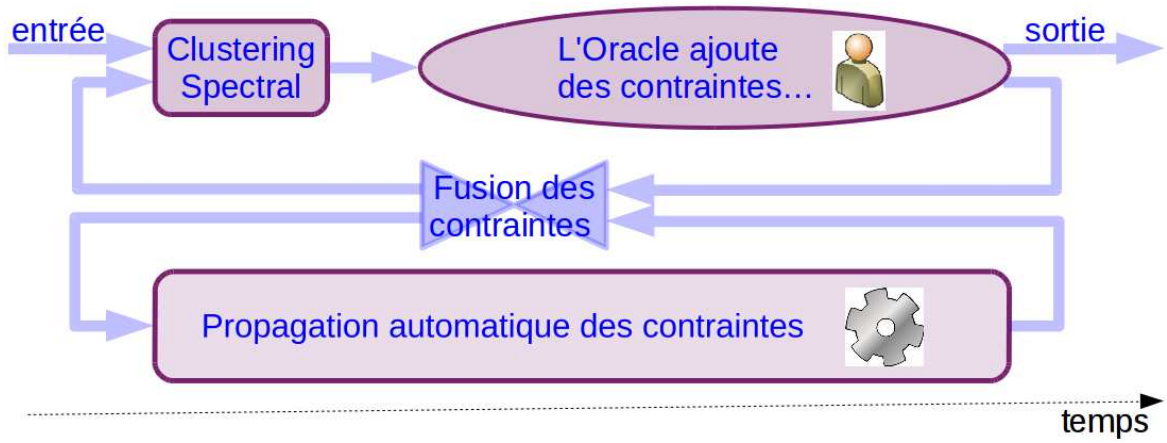


FIGURE 5.29 – Un processus optimisé effectuant en parallèle l'annotation humaine et la propagation automatique des contraintes.

Dans ce processus parallèle, l'une des deux options suivantes de synchronisation des tâches doit être choisie :

1. *La propagation automatique est exécutée totalement.* L'expert peut alors se retrouver en situation d'attente si cette propagation est longue. Cependant le processus est toujours optimisé par cette propagation totale et notre expert demeure un Oracle.

2. *La propagation automatique est interrompue dès que l'expert est disponible.* Cette propagation partielle peut impliquer une augmentation du nombre de supervisions nécessaire pour obtenir la même convergence. L'autre inconvénient est que l'on peut créer des incohérences dans la vérité terrain en proposant à l'expert de superviser des paires qui auraient pu être déduites par propagation. L'avantage est que le processus de propagation est totalement invisible pour l'opérateur humain.

Il s'agit d'un choix à effectuer entre qualité et performance. Cependant il demeure toujours l'option intermédiaire qui consiste à donner un délai d'exécution supplémentaire à la propagation après la fin de la supervision. Il faut juste régler le délai d'interruption de la propagation sur une valeur correspondant au temps d'attente supportable par l'opérateur. Ceci peut permettre à la propagation d'aller plus loin, voire même d'être complète. Cependant cette option intermédiaire conserve les mêmes avantages et inconvénients que la deuxième option car la propagation peut aussi être interrompue et donc incomplète.

5.3.6 Impact de la propagation sur la prise en compte des contraintes de la méthode COSC

Nous proposons dans cette section d'observer le gain apporté par la propagation sur différentes méthodes de Clustering Spectral.

5.3.6.1 Problématique

La méthode COSC est présentée dans le papier [Rangapuram et Hein, 2012]. Elle est validée avec les jeux de données du Centre pour le Machine Learning et les Systèmes Intelligents (CMLIS) de l'Université de Californie à Irvine (UCI). La figure 5.30 présente la comparaison de la méthode COSC sur le jeu de données « Sonar » avec les méthodes de l'état de l'art du Clustering Spectral semi-supervisé selon un schéma actif par sélection aléatoire. Ces com-

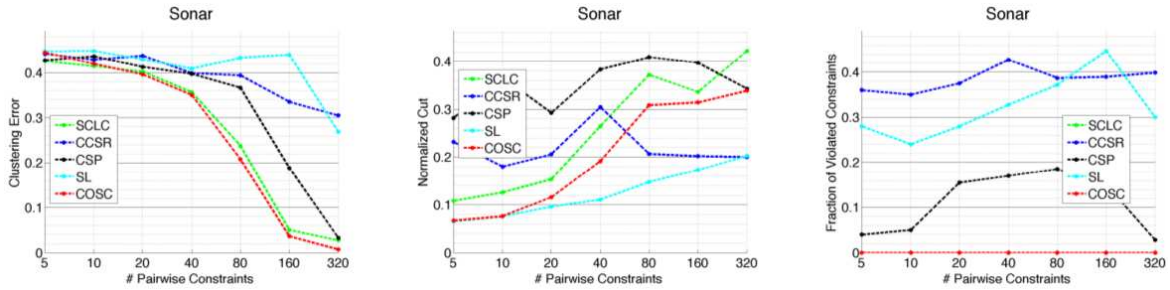


FIGURE 5.30 – Comparaison de la méthode COSC aux d'autres méthodes de l'état de l'art sur le jeu de données « Sonar » du CMLIS de l'UCI [Rangapuram et Hein, 2012].

paraisons montrent que COSC est la meilleure des méthodes et que la moins bonne est le Spectral Learning (SL). Ce qui semble tout à fait logique, car COSC intègre les contraintes au Clustering Spectral selon un procédé d'optimisation convexe sophistiqué, tandis que SL cherche juste à contraindre le Clustering Spectral de manière simple en ajoutant des 1 et des 0 dans la matrice de similarité.

Il faut bien noter que le jeu de données « Sonar » est composé de 208 objets répartis en 2 classes et nous remarquons que les comparaisons sont menées jusqu'à 320 paires. Au bout de ces 320 paires choisies aléatoirement, aucune méthode n'atteint un clustering parfait même

si COSC s'en rapproche. Or nous savons que si l'on ajoute à une méthode de Clustering Spectral semi-supervisé actif avec sélection aléatoire des paires, une étape de propagation automatique des contraintes, sur un jeu de données bi-classe de n objets, on a la garantie d'obtenir le clustering parfait en n itérations (dans notre cas $n = 208$).

5.3.6.2 Impact de la propagation entre des méthodes de Clustering Spectral comparables

Nous savons qu'il existe une multitude de variations autour du Clustering Spectral (choix du laplacien, calcul de la matrice d'adjacence, nombre de plus proches voisins, pondérations...). Nous voulons comparer les deux principales façon d'incorporer la connaissance (dans la matrice d'adjacence ou dans le problème spectral) mais en utilisant exactement le même algorithme de Clustering Spectral. Nous testons donc la méthode COSC sur le jeu de données « Sonar » en comparant : la méthode COSC qui prend en compte les contraintes par optimisation convexe lors de la résolution du problème spectral à exactement la même méthode de Clustering Spectral mais avec une prise en compte des contraintes à la manière du SL, dans la matrice d'adjacence. Dans ce deuxième cas, l'algorithme COSC est appelé en ne passant en paramètres aucune contrainte au problème spectral mais une matrice d'adjacence qui inclut les contraintes ML et CL par l'ajout de 1 et 0.

Nous nommons alors :

- COSC, l'algorithme COSC utilisé de manière classique,
- SL_{COSC}, l'algorithme COSC utilisé à la façon SL.

La figure 5.31 illustre la comparaison entre les deux méthodes COSC et SL_{COSC}. Les courbes montrent l'évolution de l'indice de Rand normalisé en fonction de l'augmentation du nombre de paires supervisées. Les tracés sont chacun la moyenne de 20 exécutions différentes.

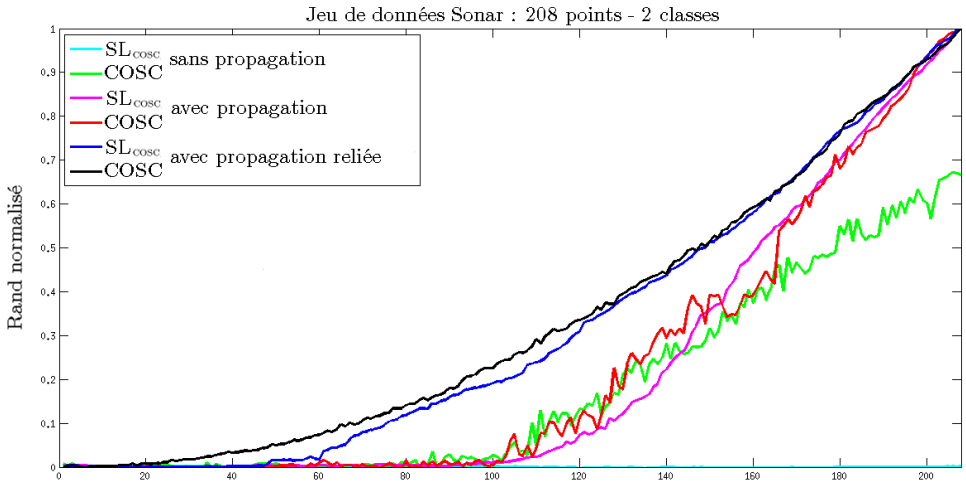


FIGURE 5.31 – Comparaison des méthodes COSC et SL_{COSC} sur le jeu de données « Sonar » de l'UCI avec ou sans propagation avec stratégie de sélection aléatoire reliée ou non.

5.3.6.3 Résultats

Nous voulons principalement savoir si la propagation automatique des contraintes, avec ou sans la stratégie de sélection des paires présentée au paragraphe 5.3.5.1, améliore la méthode SL_{COSC} jusqu'aux performances de la méthode COSC.

La courbes de couleur cyan (SL_{COSC} sans propagation) montre que SL_{COSC} ne donne pas de meilleurs résultats avec 208 contraintes injectées que sans. Le Rand normalisé reste à une valeur proche de zéro. On peut donc conclure que la prise en compte des 208 contraintes par l'ajout de 0 et 1 dans les 21 528 valeurs de la matrice d'adjacence n'est pas suffisant pour influer sur la coupe du clustering. Les contraintes ne sont pas prises en compte dans ce cas. Par contre, avec la courbe de couleur verte (COSC sans propagation), on voit qu'au bout de 208 contraintes injectées dans la méthode COSC, l'indice de Rand normalisé atteint une valeur de $0,57$. Cette valeur est bien meilleure que celle de la méthode SL_{COSC} car COSC respecte toutes les contraintes qui sont cohérentes dans ce jeu de données. Par contre la valeur reste inférieure à 1 (le clustering n'est pas parfait) car les 208 paires supervisées n'ont pas été choisies de manière optimale. On a demandé à l'expert des supervisions qui auraient pu être déduites automatiquement.

Les courbes de couleurs magenta et rouge (SL_{COSC} et COSC avec propagation totale après chaque paire supervisée) montrent qu'aucune des deux méthodes ne se détache clairement même si COSC semble donner de meilleurs résultats entre 100 et 150 paires supervisées. On note aussi que la méthode COSC présente une courbe plus instable avec des variations ponctuellement fortes. On peut expliquer ceci par le fait que certaines contraintes peuvent porter beaucoup plus d'information que d'autres. Et comme COSC respecte toutes les contraintes, en respectant une contrainte fortement informative de plus, le partitionnement peut beaucoup changer.

En comparant les courbes magenta et cyan, on peut conclure que la méthode SL_{COSC} est fortement améliorée par la propagation. La propagation accentue le respect des contraintes par cette méthode.

En comparant les courbes vertes et rouges, on peut remarquer qu'avec ce jeu de données, la propagation n'apporte pas de grande amélioration à la méthode COSC avant un minimum de 160 paires supervisées. Ceci semble s'explique par le fait que COSC respecte les contraintes et n'est pas sensible au deuxième bénéfice de la propagation. Il est seulement amélioré par la non sollicitation inutile de l'expert.

Les courbes de couleur bleue et noire (SL_{COSC} et COSC avec propagation totale après chaque paire supervisée et stratégie de sélection aléatoire reliée) sont assez proches mais la méthode COSC donne sensiblement de meilleurs résultats avec de faibles nombres de supervisions. Ceci s'explique toujours par le fait que COSC respecte toutes les contraintes contrairement à SL_{COSC}.

En comparant les courbes rouge et noire, on peut apprécier l'amélioration apportée par la stratégie de sélection aléatoire reliée sur la méthode COSC. Au bout de 100 paires supervisées, la stratégie de sélection aléatoire reliée permet d'obtenir un indice de Rand normalisé supérieur à $0,2$ alors que sans cette stratégie la qualité est toujours quasiment nulle. En comparant les courbes magenta et bleue, nous constatons la même chose avec la méthode SL_{COSC}. Ceci valide bien l'intérêt de la stratégie de sélection aléatoire reliée.

En conclusion, nous avons montré que la propagation automatique et la stratégie de sélection reliée améliore de manière significative les deux méthodes. Mais surtout, nous avons montré que la propagation automatique des contraintes amène les performances de la méthode

peu coûteuse SL_{COSC} quasiment au même niveau que celles de la méthode COSC. L'utilisation de la propagation automatique des contraintes est donc indispensable et son coût de calcul peut être compensé par l'usage d'un outil de clustering plus léger.

5.3.7 Bilan

Nous avons présenté une généralisation de la propagation automatique des contraintes utilisée dans le cas d'optimisations du clustering sur des graphes de similarités. Nous avons mené nos expérimentations sur un ensemble de données synthétiques et sur un ensemble de données réelles issues d'une classification par genre de vidéos. Nous avons montré les bénéfices de la généralisation de la propagation automatique sur le clustering de telles données. Ces gains ont été mis en évidence avec deux techniques différentes du Clustering Spectral semi-supervisé.

La principale contribution réside dans la généralisation de la propagation automatique des contraintes. Elle réduit le coût d'ajout des contraintes en améliorant la qualité du clustering obtenu. Et dans le pire des cas, les performances ne sont pas inférieures à celles des méthodes d'origine. L'utilisation de la propagation automatique des contraintes apparaît donc comme indispensable.

Nous avons aussi vu que l'introduction de la propagation revisite les comparaisons entre les méthodes de l'état de l'art. Avec la propagation automatique des contraintes, un algorithme de « basse qualité » mais à coût de calcul réduit comme le Spectral Learning permet d'obtenir quasiment les mêmes performances qu'un algorithme fiable et lourd comme COSC. Il faut donc en tenir compte lorsque l'on doit choisir une méthode de Clustering Spectral semi-supervisé.

Nous avons aussi vu qu'il reste des améliorations à faire autour des stratégies de choix des paires à superviser [Xiong *et al.*, 2014], [Vu *et al.*, 2012]. Nous avons présenté une stratégie simple amplifiant les propagations. Nous avons validé cette stratégie avec l'algorithme COSC sur des jeux de données issus de l'UCI.

Nous allons continuer nos travaux sur l'aspect actif du Clustering Spectral semi-supervisé. Nous voulons étudier les différentes manières de sélectionner les paires à superviser. Nous avons comme objectif d'étudier la combinaison des différentes stratégies de sélection des paires à superviser entre elles et avec la stratégie de sélection aléatoire reliée.

Une autre perspective est d'intégrer d'autres méthodes de classification comme le Deep Learning dans des processus itératifs voire actifs. Et nous voulons étudier l'ajout de la propagation automatique des contraintes dans ces méthodes.

Chapitre 6

Conclusions et Perspectives

6.1 Conclusions

Ce travail de thèse intitulé « Structuration de bases multimédia pour une exploration visuelle » commence par une étude de la visualisation de bases multimédia. Après avoir examiné la visualisation d'information et les visualisations existantes bien décrites dans la littérature, nous avons constaté qu'un processus de visualisation nécessite plusieurs étapes intermédiaires pour passer d'une base multimédia brute à une visualisation exploitable pour l'exploration des média. Nous avons donc produit un modèle décrivant ce processus de visualisation. Dans ce processus, il existe des aspects directement liés à la visualisation qui concernent l'ergonomie IHM, domaine spécifique qui n'a pas fait l'objet de notre étude. Sur un plan plus amont, nous avons aussi mis en évidence que les processus de visualisation nécessitent une structure adaptée des données et nous avons alors orienté tous nos travaux sur cette problématique de structuration d'une base multimédia.

Nous avons commencé par montrer que la description de la structuration de ce modèle de visualisation est intéressante : elle permet de tracer les différentes étapes des méthodes de structuration. En examinant ces méthodes, nous avons constaté que la première étape de structuration consiste en l'extraction des descripteurs à partir des données brutes. Pour ce qui concerne l'extraction des descripteurs bas niveau à partir d'une base de film, nous avons simplement réutilisé les résultats de travaux antérieurs qui nous ont directement donné les descripteurs. Notre travail de structuration commence donc après cette première étape avec des descripteurs des données déjà délivrés. Nous avons alors mis au point différentes techniques de structuration qui en fonction de la nature des descripteurs disponibles permettent d'obtenir efficacement la structure finale désirée pour qu'elle puisse être représentée et visualisée.

La première structuration que nous avons envisagée est basée sur les corrélations de rang. Nous avons exploré cette approche dans le cas d'une base de données multimédia constituée de films. Sur cette base particulière nous voulions disposer d'une mesure de ressemblance entre films qui soit proche d'annotations de l'opinion humaine afin d'envisager par exemple une navigation de proche en proche personnalisable. Cette base est fournie avec des métadonnées décrivant les films de manière globale et pour laquelle nous savons extraire des descripteurs numériques des données bas niveau sémantique. Les films sont de courte durée et suffisamment homogènes pour envisager la production d'une mesure de ressemblance non

pas entre les scènes mais entre les films entiers. Notre première contribution a consisté en la proposition d'une nouvelle mesure de dissimilarité automatique entre films reproduisant l'opinion humaine. Cette solution est basée sur une fusion entre les dissimilarités bas niveau et métadonnées. Comme l'agrégation d'information de différents types ne peut être réalisée de façon numérique, nous avons proposé une solution originale basée sur les coefficients de corrélation de rang. Nous avons validé que la dissimilarité produite classe bien en premier les vidéos les plus ressemblantes. Le principal intérêt de la méthode est d'être automatique en ne nécessitant ni normalisation des données, ni ajustement de paramètres. Cette mesure de dissimilarité automatique pourrait aussi être utilisée dans une visualisation pour aider l'utilisateur dans sa sélection de films en lui proposant les films qu'il pourrait apprécier.

Ensuite, nous avons adapté ces corrélations pour produire une méthode de classification de base multimédia permettant de regrouper ensemble des données ressemblantes afin de pouvoir les visualiser. Cette méthode est elle aussi automatique, ne nécessitant ni normalisation des données, ni ajustement de paramètres. Nous l'avons appliquée aux données d'un challenge de classification consistant en la reconnaissance du genre de séquences vidéos pour lesquelles nous disposons des descripteurs (image, son et texte) sous forme de valeurs numériques. Cependant avec cette méthode de classification nous obtenons des performances qui ne sont pas meilleures que les autres méthodes de l'état de l'art. L'avantage principal de la méthode reste donc la prise en compte de descripteurs de type différents sans contrainte de normalisation. Nous avons donc décidé d'explorer ensuite une méthode de classification particulière : le Clustering Spectral.

Nous nous sommes intéressés en particulier à l'ajout de connaissance pour rendre les techniques du Clustering Spectral supervisées ou semi-supervisées. Notre premier travail a été de tester le Clustering Spectral dans un cadre supervisé et de le confronter aux méthodes de l'état de l'art. Les résultats ont montré que dans ce cadre le Clustering Spectral ne pouvait pas rivaliser avec les meilleures méthodes de classification supervisées. Nous nous sommes alors dirigé vers les méthodes de Clustering Spectral semi-supervisé interactives. Après avoir examiné les méthodes de l'état de l'art et avoir analysé les pistes de réduction des coûts de collecte d'annotation pour la supervision, nous nous sommes focalisés sur la propagation automatique de contraintes de similarité. Notre première contribution a porté sur la généralisation des propagations automatiques. Nous avons validé l'intérêt et les bénéfices apportés par cette généralisation. Nous avons aussi vu que l'introduction de la propagation revisite les comparaisons entre les méthodes de l'état de l'art. Nous avons montré que grâce à la propagation automatique des contraintes, un algorithme de Clustering Spectral « basse qualité » mais à coût de calcul réduit permet d'obtenir quasiment les mêmes performances qu'un algorithme fiable mais à grand coût de calcul. Il faut donc en tenir compte lorsque l'on doit choisir une méthode de Clustering Spectral semi-supervisée et donc préférer utiliser un algorithme de clustering peu coûteux mettant à profit la propagation automatique généralisée des contraintes.

6.2 Perspectives et travaux futurs

L'ensemble de ces travaux ouvre différentes perspectives. Certaines à court terme et en lien direct avec les travaux présentés ont déjà été évoquées en fin des sections aux chapitres

précédents. Mais ces perspectives peuvent être élargies à d'autres travaux.

Comme la majorité de nos travaux ont portés sur des méthodes de structuration par classification, nous avons réalisé un prototype de visualisation 3D et d'annotation. Il permet d'explorer et de classer une base de vidéos regroupées par clusters inclus dans des ellipsoïdes et sphéroïdes. Nous pourrions envisager de revenir sur cette visualisation et produire une application aboutie qui permette l'interactivité de la classification semi-supervisée.

Les traitements efficaces s'appuient sur des données multimodales (image, son ou texte). La solution envisagée est souvent une fusion des données dès le début du processus. C'est cette stratégie que nous avons d'ailleurs employée. Une stratégie alternative consiste à fusionner les données plus tard dans le traitement. Dans ce schéma, l'utilisation d'hypergraphes offrirait une extension naturelle de l'approche de Clustering Spectral que nous avons proposé.

Nous avons aussi vu qu'il reste des améliorations à faire autour des stratégies de choix des paires à superviser. Nous avons présenté une stratégie simple amplifiant les propagations. Nous pouvons continuer nos travaux sur l'aspect actif du Clustering Spectral semi-supervisé en étudiant les différentes manières de sélectionner les paires à superviser. Il serait intéressant d'étudier si la propagation automatique des contraintes permet d'améliorer ces stratégies et d'étudier la combinaison des différentes stratégies de sélection des paires à superviser entre elles ou avec la stratégie de sélection aléatoire reliée.

Pour finir, le travail qui va certainement nous intéresser le plus est d'intégrer d'autres méthodes de classification prometteuses comme le Deep Learning dans des processus itératifs voire actifs et d'étudier l'apport de la propagation automatique des contraintes dans ces méthodes. Ces travaux ainsi que tous les autres pourront être étendus au delà des films et des images, par exemple aux données satellitaires. Ce domaine en particulier a ouvert très récemment ses données en accès libre. Les besoins d'analyse de données dans ce secteur sont nombreux, portant sur la gestion des crises humanitaires jusqu'à l'optimisation des territoires et l'écologie. Les méthodes proposées s'adapteraient bien à cet univers de données massives encore faiblement annotées.

Annexe A

Les visualisations existantes

Dans cette étude des visualisations existantes, nous nous intéressons aussi bien aux vidéos qu'aux images car les vidéos sont couramment figurées par une image clé. Cette image peut être l'affiche du film, la première image de la vidéo ou un assemblage d'images clés. Nous nous intéressons aussi aux autres données multimédia (son et texte) lorsque les techniques employées sont applicables aux vidéos.

Les visualisations sont classées en suivant les trois niveaux : structures, représentations et visualisations données à l'espace brut formé par la base documentaire. Le plan du premier niveau correspondant à la structuration et il respecte la classification proposée au paragraphe 2.4.

A.1 Structure orientée valeurs

Les structures orientées valeurs sont les relations $E \times F$ où E est notre base documentaire et F un ensemble de valeurs. Cette structuration orientée valeurs consiste simplement à affecter des valeurs à chaque élément de la base E.

A.1.1 Structure unidimensionnelle

A.1.1.1 Cas général

Il s'agit d'une structure orientée valeurs obtenue en associant à chaque média une valeur numérique réelle. La base documentaire a donc la puissance du continu et elle peut être représentée sur la droite réelle.

Si la valeur numérique est importante pour l'exploration, la visualisation peut la faire figurer de manière explicite. C'est le cas des chronologies comme l'exemple donné en figure A.1.

Cependant, cette structure continue, trop générale, n'est que très peu représentée et visualisée. Elle est surtout utilisée comme intermédiaire avant l'obtention d'une structure discrète.

A.1.1.2 Structure discrète

Elle repose sur une bijection entre les N_b média de la base documentaire et l'intervalle entier $[\![1; N_b]\!]$. En ce sens, cette structure de type liste ordonnée est un cas particulier de la structure orientée valeurs unidimensionnelle. Mais cette structure de type liste chaînée est

FIGURE A.1 – Exemple de chronologie (http://TimeRime.com).

aussi une structure orientée liaisons. Sa relation « a pour suivant » rend l'espace totalement ordonné.

Elle permet de nombreuses représentations et visualisations, pour des explorations simples. Les visualisations induites évitent les problèmes d'occlusion. Cependant pour trouver un média bien précis, l'exploration peut parfois être longue. Classiquement les moteurs de recherche calculent un indice de « pertinence » pour chaque média. Ils créent ensuite la liste ordonnée selon ce critère. L'algorithme PageRank [Brin et Page, 1998] est l'idée principale du classement effectué par Google [Eisermann, 2008].

Contrairement aux moteurs de recherche, de nombreuses applications créent cette structure en liste de manière aléatoire ou sans aucune intention particulière (par exemple, l'utilisation de l'identifiant unique de la base de donnée). L'intérêt de ces applications repose sur leurs représentations et visualisations.

La représentation type pellicule 2D est une représentation discrète beaucoup utilisée. Il s'agit d'une grille composée d'une seule ligne avec un premier et un dernier élément comme représenté dans la figure A.2.

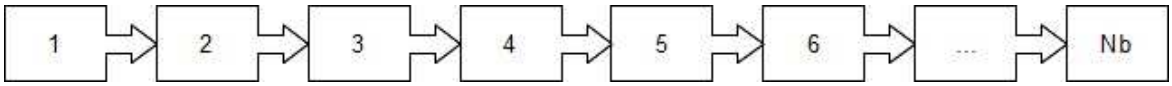


FIGURE A.2 – Pellicule 2D.

FIGURE A.3 – Logiciel AutoViewer (http://www.simpleviewer.net).

Le logiciel AutoViewer de la société SimpleViewer, présenté en figure A.3, propose une visualisation uniforme, partielle et défilante purement de type pellicule 2D.

L'aperçu d'images de l'explorateur Microsoft Windows (figure A.4), comme certains logiciels type appareils photo proposent aussi ce type de visualisation en pellicule 2D, couplée

avec une vue agrandie de l'élément courant.

FIGURE A.4 – Aperçu d'images de l'explorateur Microsoft Windows XP et 7.

Il existe aussi des visualisations non uniformes type FishEye comme le menu Mac OS X (figure A.5).

FIGURE A.5 – Menu Mac OS X de type FishEye.

Les représentations type pellicule 3D sont une extension des représentations type pellicule 2D que l'on rencontre fréquemment.

Les interfaces Cover Flow d'Apple (figure A.6) et le Flip 3D de Microsoft Windows (figure A.7) proposent une visualisation déformante tridimensionnelle.

FIGURE A.6 – Cover Flow d'Apple.

FIGURE A.7 – Flip 3D de Microsoft.

Vangelis Pappas-Katsiafas (http://www.katsiafas.com/visualperception) propose de nombreuses représentations et visualisations en 3D dans la figure A.8.

Les listes circulaires sont une extension des représentations type pellicule qui consistent en une liste dont le « dernier » élément pointe sur le « premier ». Elles sont couramment représentées de manière circulaire en 2D ou 3D. De nombreux scripts ou logiciels proposent des visionneuses d'images sous forme de carrousel 3D. La figure A.9 présente un plugin jQuery permettant d'afficher un ensemble d'images sous la forme d'un carrousel 3D.

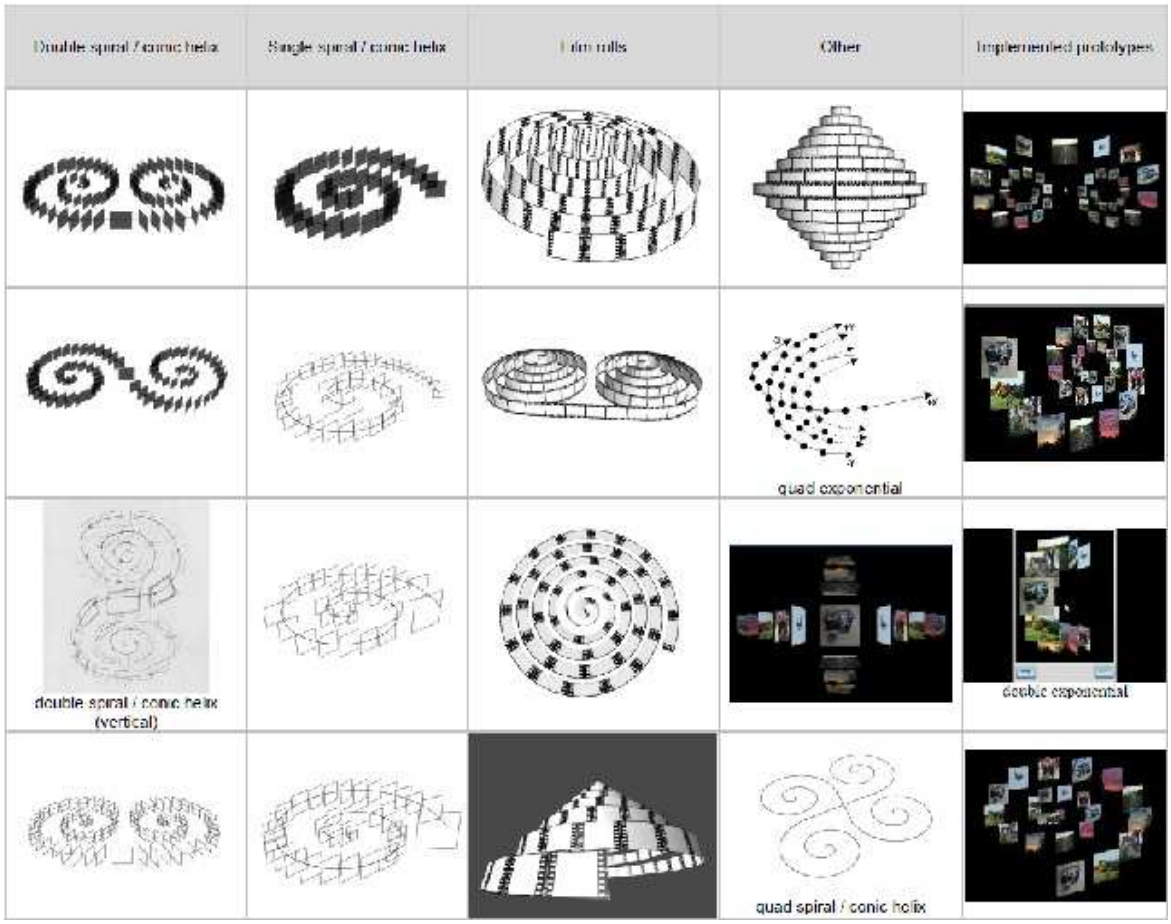


FIGURE A.8 – Représentations et visualisations 3D de Vangelis Pappas-Katsiafas.

FIGURE A.9 – Plugin jQuery (« 3D Carousel »).

Les représentations en grille sont aussi une représentation des structures discrètes que l'on rencontre fréquemment.

L'ordonnancement sur une grille peut être effectué de plusieurs façons. Le plus courant est de gauche à droite puis de haut en bas comme représenté dans la figure A.10. C'est la représentation classique des moteurs de recherche comme Google (figure A.11). Ils proposent une visualisation uniforme découpée en plusieurs pages avec une navigation entre page.

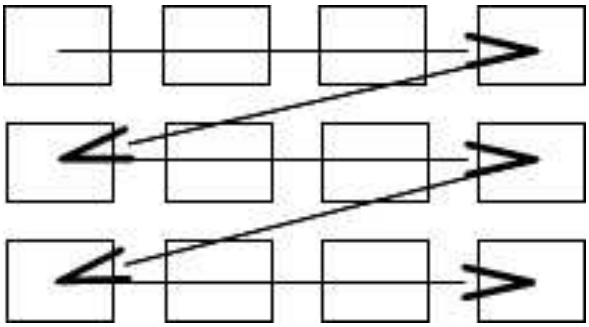


FIGURE A.10 – Représentation en grille d'un structure unidimensionnelle.

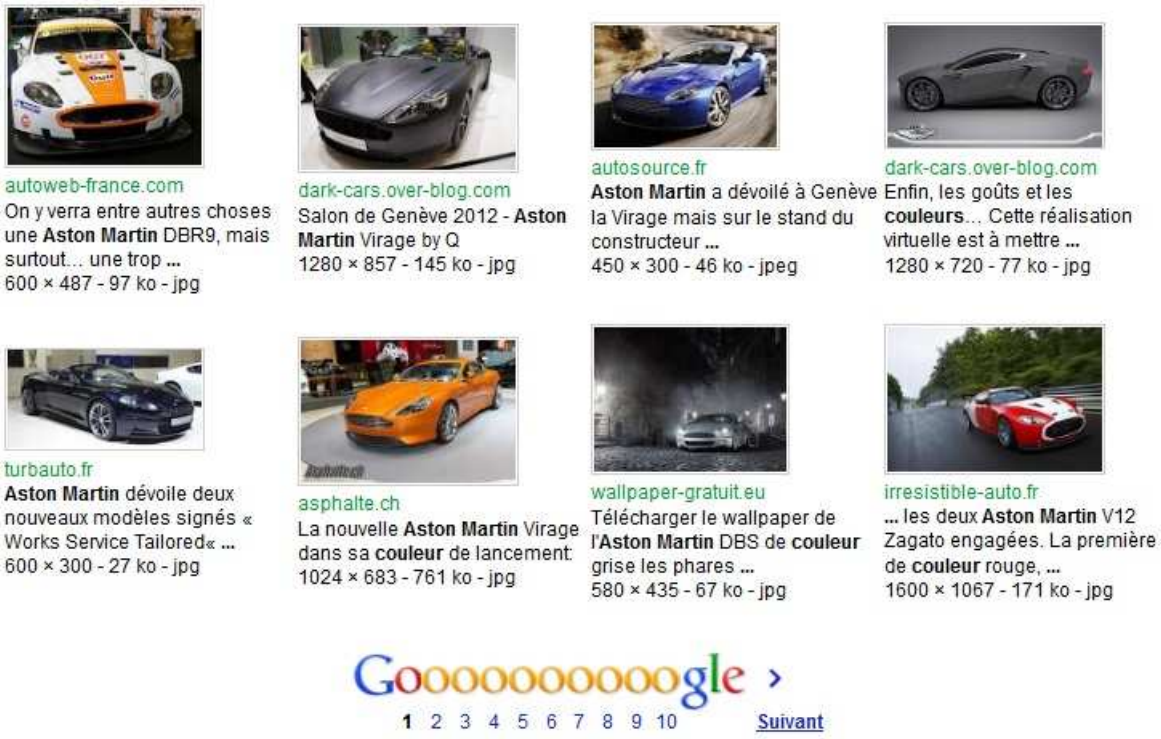


FIGURE A.11 – Recherche d'image avec Google.

Comme les pellicules, la grille peut être visualisée de manière défilante. Le plugin PicLens (figure A.12) propose une visualisation avec un défilement 3D basé sur une déformation monofocale unidirectionnelle de type « mur fuyant ».

En synthèse, pour obtenir toutes ces structures unidimensionnelles décrites précédemment, il faut ordonner les données en fonction d'une requête. En général, les besoins de structuration reposent sur la production d'une métrique et l'utilisation d'algorithmes de classement.

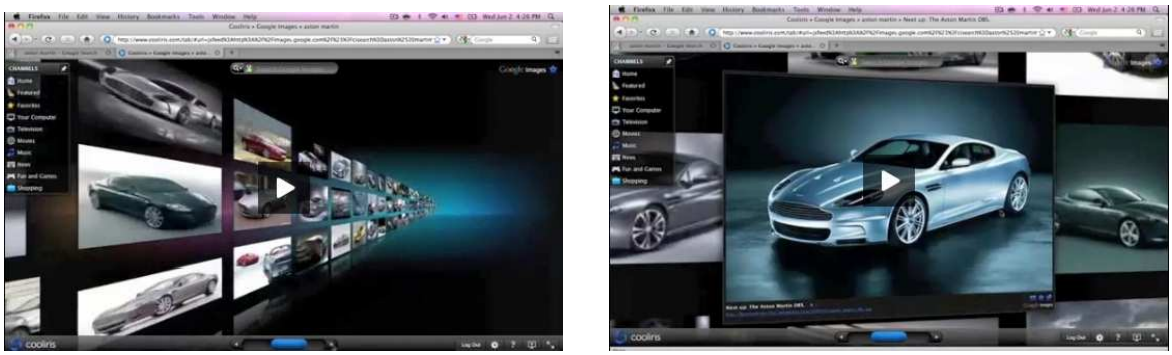


FIGURE A.12 – Plugin PicLens de la société Cooliris (http://www.cooliris.com).

A.1.2 Structure bidimensionnelle

Il s'agit d'une structure orientée valeurs obtenue en associant à chaque média deux valeurs numériques. La représentation naturelle est la représentation graphique 2D qui présente souvent l'inconvénient de la superposition des documents. En voici quelques exemples.

Oskope (figure A.13) est un outil de recherche visuelle développé par la société oShope media gmbh. Il dispose de plusieurs vues dont une basée sur une représentation graphique 2D où les images sont par exemple placées en fonction de leur rang et taille.

FIGURE A.13 – Outil de recherche visuelle Oskope (http://www.oskope.com).

Flickr de Yahoo (http://www.flickr.com) est un service de gestion et de partage de photos et vidéos en ligne. Flickr s'est fixé deux objectifs principaux : permettre aux utilisateurs de partager leurs photos et vidéos avec les personnes de leur choix et proposer de nouvelles méthodes d'organisation. Parmi ces méthodes, il est possible d'ajouter les média à une carte géographique comme l'exemple de la figure A.14. La carte peut être bidimensionnelle avec pour coordonnées les geotags latitude et longitude. Les photos peuvent aussi être explorées dans l'interface tridimensionnelle GoogleEarth avec l'ajout de tags supplémentaires indiquant le point de vue.

La visualisation présentée au paragraphe 2.4 est un autre exemple illustrant les struc-

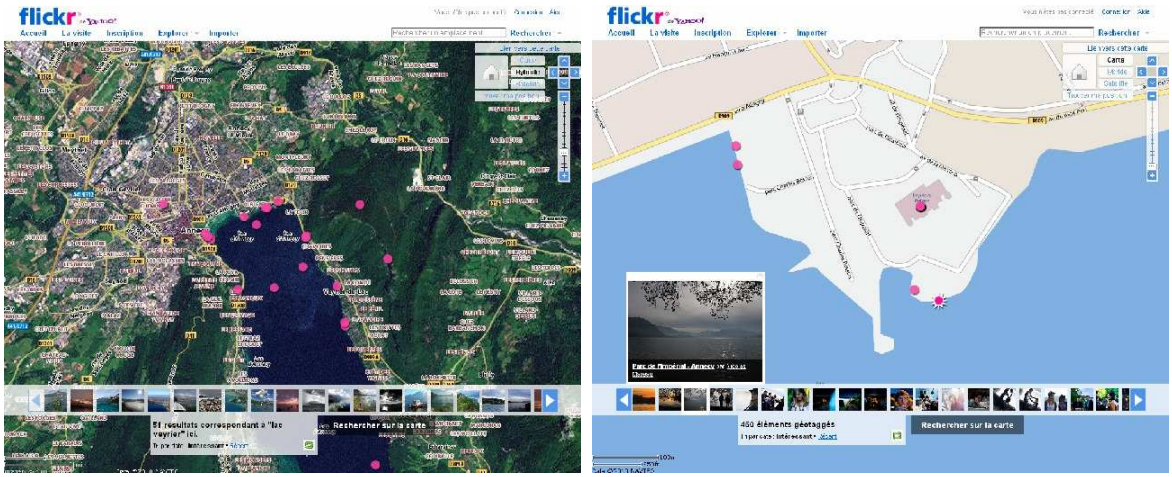


FIGURE A.14 – Organisation géographique de média Flickr.

tures bidimensionnelles. Cette visualisation utilise une technique MDS « Multi Dimensional Scaling » itérative complétée par une visualisation de type Fisheye sur une représentation en grille 2D [Liu *et al.*, 2004].

En synthèse, pour obtenir ces placements bidimensionnels, le processus de structuration nécessite généralement une méthode de projection.

A.1.3 Structure de grande dimension $E \times \mathbb{R}^n$

Il s'agit d'une structure orientée valeurs obtenue en associant à chaque média n valeurs numériques. La représentation naturelle serait une représentation graphique dans $\mathbb{R}^n$. Cependant cette représentation est difficilement visualisable. Il est possible d'envisager une visualisation locale centrée sur un document avec une projection 2D ou 3D de ses (plus proches) voisins.

Maximo [Maximo *et al.*, 2009] présente le M-Cube : un outil de visualisation pour les bases de données multimédia multidimensionnelles. Un M-Cube est un cube 3D dans lequel sont placés des média selon 3 descripteurs. Des descripteurs supplémentaires peuvent être représentés en utilisant la taille des icônes, leur couleur... Les descripteurs non représentés sont proposés sur les 3 axes géométriques. Lorsqu'un descripteur est substitué à un autre, la projection est actualisée. Dans le M-Cube de gauche de la figure A.15, les 3 descripteurs « artist », « year » et « location » donnent le positionnement 3D de l'icône dans le M-Cube. Les descripteurs « theme » et « rating » sont proposés en dessous de chacun des 3 axes pour être substitués à l'un des 3 descripteurs précédents. Le descripteur « filetype » est représenté par la couleur de l'icône. Le descripteur filesize est représenté par la taille de l'icône. Le M-Cube peut aussi subir des opérations de rotation, zoom et filtrage. Dans une base de données vidéo, lorsqu'un média est sélectionné, la lecture démarre comme représenté dans le M-Cube de droite de la figure A.15.

Ces structures orientées valeurs de grande dimension ne nécessitent pas de méthode de structuration complexe. La visualisation consiste simplement en une sélection des descripteurs à afficher. En général, l'extraction des descripteurs est la seule étape de structuration nécessaire.

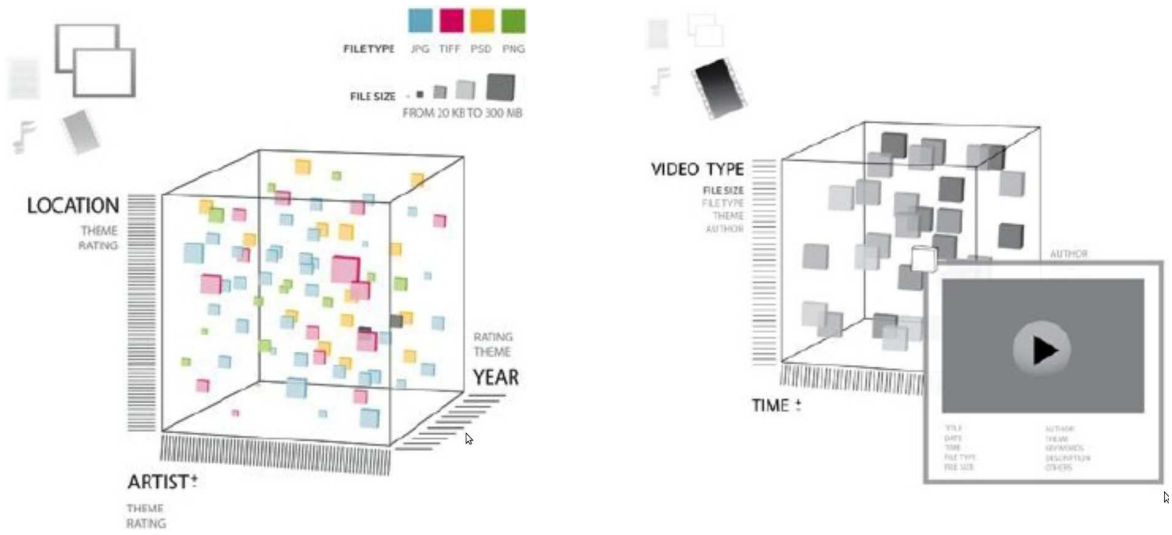


FIGURE A.15 – Deux exemples de M-Cube. A gauche, sont représentés des média selon 7 descripteurs (artist, year, location, theme, rating, filetype et filesize). A droite, la visualisation de vidéo par sélection d'une icône.

A.1.4 Structure de grande dimension $E \times F^n$ avec $F \neq \mathbb{R}$

Ce ne sont plus comme au paragraphe précédent, des valeurs numériques qui sont associées aux média, mais par exemple des mots-clés (ou tags). Tag Galaxy (figure A.16) qui a été développé par Steven Wood au cours de sa thèse en 2008, repose sur une telle structure. Tag Galaxy offre une exploration de la base Flickr avec trois niveaux complémentaires de visualisation.

- Tout d'abord, après avoir saisi un mot-clé, nous obtenons la première visualisation. La première image de la figure A.16 est obtenue avec le mot-clé Annecy. Il s'agit d'un « système solaire » centré sur le mot-clé principal Annecy. Autour sont affichés 8 autres tags : Haute-Savoie, Savoie, France, Lake, Lac, Alpes, Alps, Water. Ce sont les tags les plus présents sur toutes les images ayant le tag Annecy. Dans cette visualisation, nous pouvons sélectionner un des 8 tags proposés pour préciser le filtre à appliquer sur la base documentaire. Nous pouvons par exemple choisir Annecy, puis Alpes et Faune. Les images proposées dans la seconde visualisation correspondront au filtre : « toutes les images ayant les tags Annecy, Alpes et Faune ».

- Cette deuxième visualisation est illustrée par les trois images suivantes de la figure A.16. Elle consiste en une grille sphérique permettant de parcourir les vignettes des images associées au filtre créé dans la première vue. Chaque « page » est affichée sur une nouvelle sphère.

- La troisième visualisation est un zoom sur l'image sélectionnée comme présenté dans la dernière image de la figure A.16.

Tag Galaxy n'utilise pas de restructuration des données. Les images restent caractérisées par leur ensemble de mots-clés. Tout le mécanisme de navigation est dans la couche de visualisation. Pour chaque mot-clé choisi, les images sont positionnées arbitrairement sur une sphère.

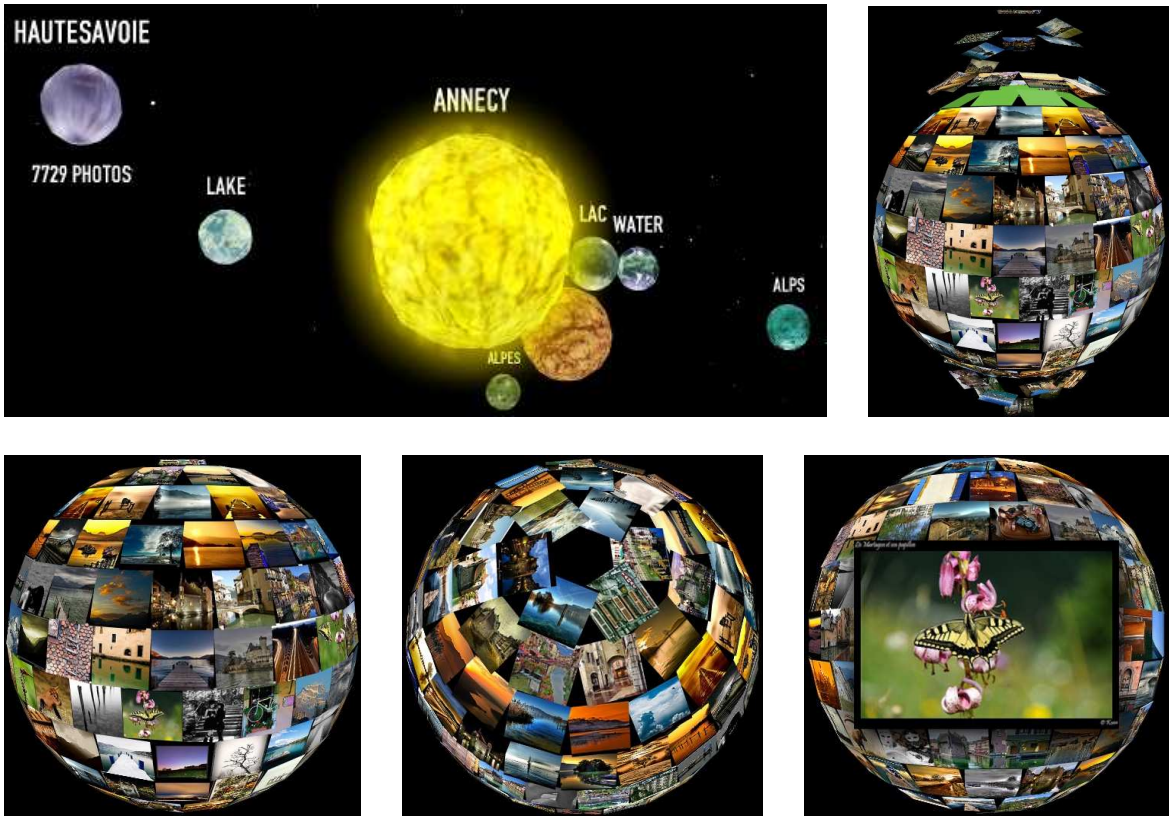


FIGURE A.16 – Navigation à l'aide de l'application Tag Galaxy (http://taggalaxy.de) dans la base d'images Flickr. Le point de départ de la navigation est le Tag « Annecy ».

A.2 Structure orientée liaisons

Les structures orientées liaisons se différencient des structures orientées valeurs simplement par le fait qu'elles mettent en relation les éléments de notre base documentaire E avec eux-mêmes.

La relation peut avoir différents objectifs : donner un parcours de navigation intelligible, mettre en évidence des clusters, créer une hiérarchie, ordonner la base...

Il faut noter que toutes les relations rencontrées sont des relations binaires et que celles-ci sont couramment représentées par des graphes. Parmi les graphes rencontrés, nous trouvons principalement des graphes non orientés quelconques et des hiérarchies. On ne rencontre pas de relations quelconques comme les relations n-aires qui pourraient être représentées par des hypergraphes.

Quant aux relations valuées, elles ne sont pas directement représentées (graphes pondérés ou matrices). Par exemple, la relation de similarité (distance entre éléments dans la dimension n) ne peut pas être représentée globalement de manière conforme. En utilisant des traitements comme des seuillages, ces relations valuées sont transformées en de nouvelles relations non valuées.

A.2.1 Les graphes non orientés quelconques

Il existe de nombreuses applications permettant d'explorer des bases documentaires sous la forme d'un parcours de graphe. Parmi celles-ci, Videosphere (figure A.17) propose une navigation dans une base de vidéos. Il s'agit d'une représentation 3D sur une sphère d'un graphe de vidéo avec possibilité de visualisation depuis l'extérieur ou l'intérieur de la sphère. La relation utilisée est basée sur une distance sémantique (nombre de tags communs entre deux vidéos). Videosphere utilise un positionnement figé des média sur la sphère. Les liens relient les média par proximité sémantique alors que ces média peuvent être éloignés sur la sphère. En conséquence, cette application ne permet pas un aperçu du voisinage du média courant. Elle ne permet pas non plus de revenir facilement au média précédent. Le principal défaut de ce genre d'application est de proposer une navigation qui est quelque peu erratique.

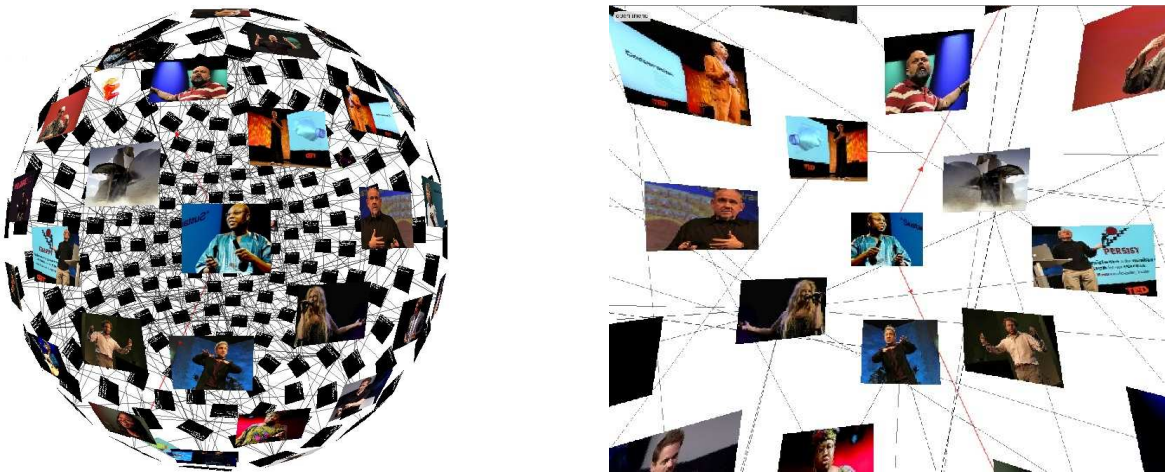


FIGURE A.17 – Videosphere (http://old.bestiario.org/research/videosphere). A gauche : vue extérieure. A droite : vue intérieure.

Afin de permettre une navigation plus intelligible, les placements radiaux sont couramment utilisés. Boutin [Boutin, 2005] présente un tel placement dans sa thèse. Ces placements naturels pour les arbres sont étendus aux graphes quelconques comme illustré en figure A.18. Pour cela, la distance entre deux nœuds est définie comme étant le nombre minimum d'arcs nécessaires pour aller d'un nœud à l'autre. Le nœud sur lequel se porte le focus est placé au centre. Tous les nœuds qui lui sont directement connectés sont placés sur la première couronne. Tous les nœuds à une distance de 2 du nœud central sont placés sur la deuxième couronne. Et ainsi de suite, jusqu'à ce que tous les nœuds du graphe soient placés. La visualisation permet de changer le nœud central et de repositionner tous les autres nœuds en conséquence. Cependant, Boutin décrit ce placement uniquement pour des nœuds indicés. Il travaille avec des graphes qui relient uniquement des points. Il ne s'intéresse pas à la problématique des nœuds visuels que sont les données multimédia. Le principal problème rencontré avec les nœuds visuels est la superposition des contenus affichés et leur manque de visibilité lorsqu'ils deviennent trop petits.

Afin de permettre une navigation intelligible avec des nœuds visuels, Jankun-Kelly [Jankun-Kelly et Ma, 2003] propose les MoireGraphs (figure A.19). Il s'agit d'un focus radial complété par un contexte de visualisation et d'interaction pour les graphes avec nœuds visuels.

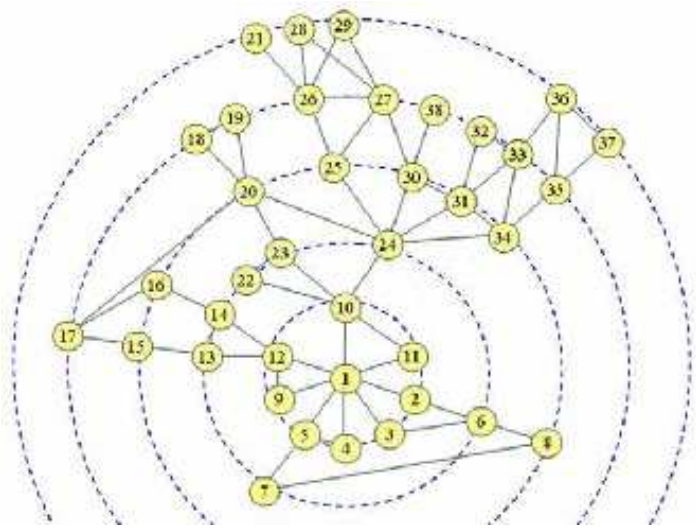


FIGURE A.18 – Placement radial d'un graphe quelconque [Boutin, 2005].

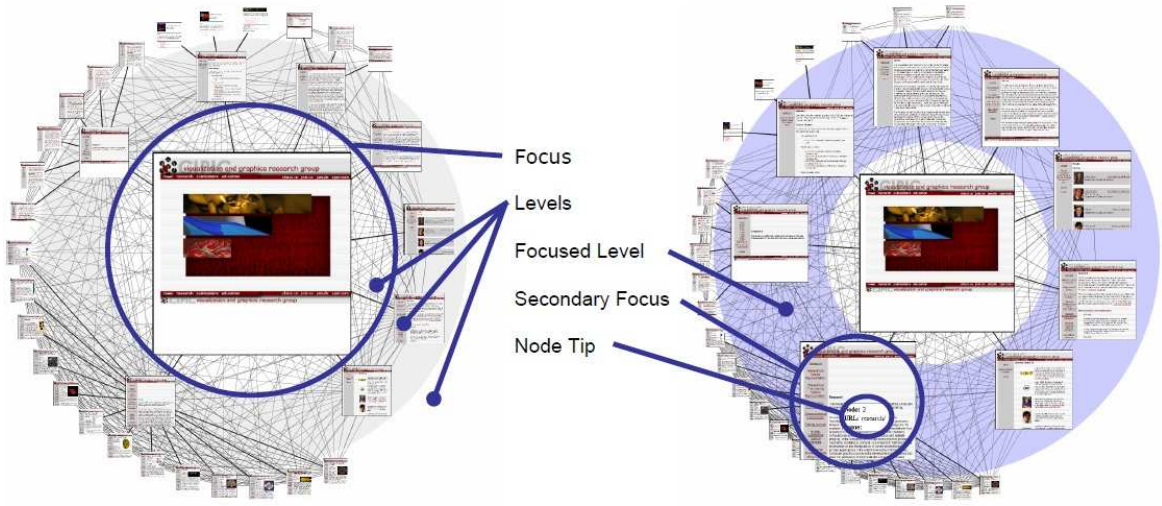


FIGURE A.19 – Les MoireGraphs proposés par Jankun-Kelly [Jankun-Kelly et Ma, 2003].

Les nœuds visuels sont inscrits dans des disques et couronnes qui ne se chevauchent pas. La taille de ces disques est déterminée par le niveau de la couronne à laquelle ils appartiennent. Le processus de navigation est totalement décrit par Jankun-Kelly [Jankun-Kelly et Ma, 2003]. Le passage d'un nœud central à un autre est fluide. Le focus peut être porté sur une couronne. Un focus secondaire peut être ciblé sur un nœud précis et les tailles des nœuds visuels sont modifiés en conséquence.

Ces visualisations de données multimédia ne nécessitent pas de restructuration des données. Les graphes sont représentés sous forme de placements en graphe développé ou radial. Ensuite, la navigation est gérée dans la couche de visualisation à l'aide des 7 tâches de la Type by Task Taxonomy.

A.2.2 Les hiérarchies

Les placements radiaux présentés précédemment sont des placements couramment utilisés et naturels pour les hiérarchies. Parmi les visualisations existantes, citons les MoireTrees ([Mohammadi-Aragh et Jankun-Kelly, 2005]) qui sont une adaptation des MoireGraphs pour visualiser des données multi-hiérarchiques. Dans la figure A.20, figurent des images d'une base de données de la NASA organisées à gauche par une hiérarchie stellaire et à droite par une hiérarchie des missions d'exploration.

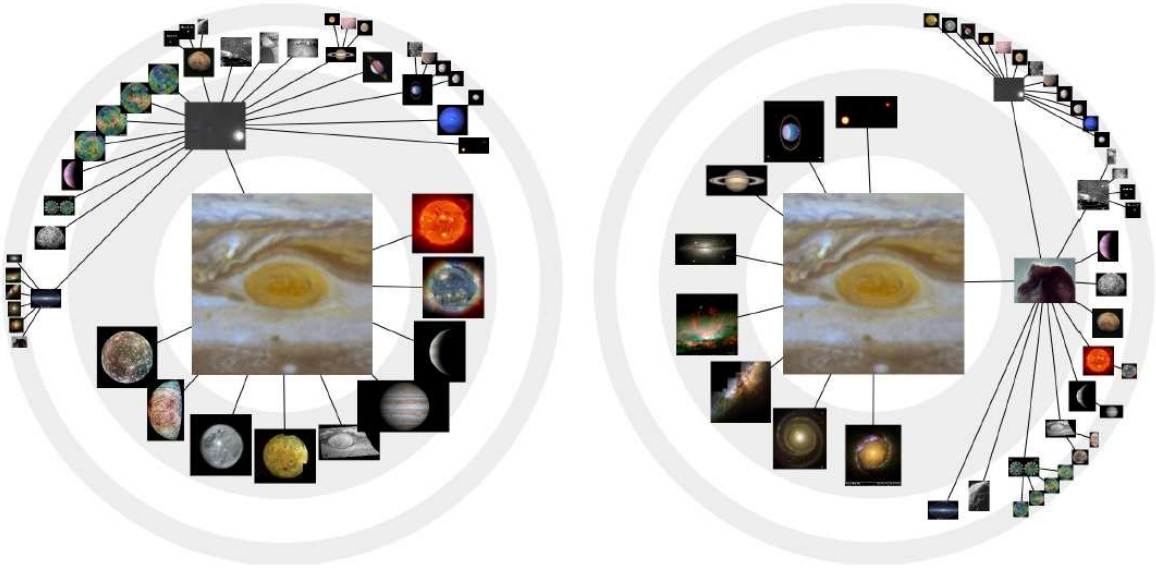


FIGURE A.20 – Les MoireTrees proposés par Mohammadi-Aragh et Jankun-Kelly.

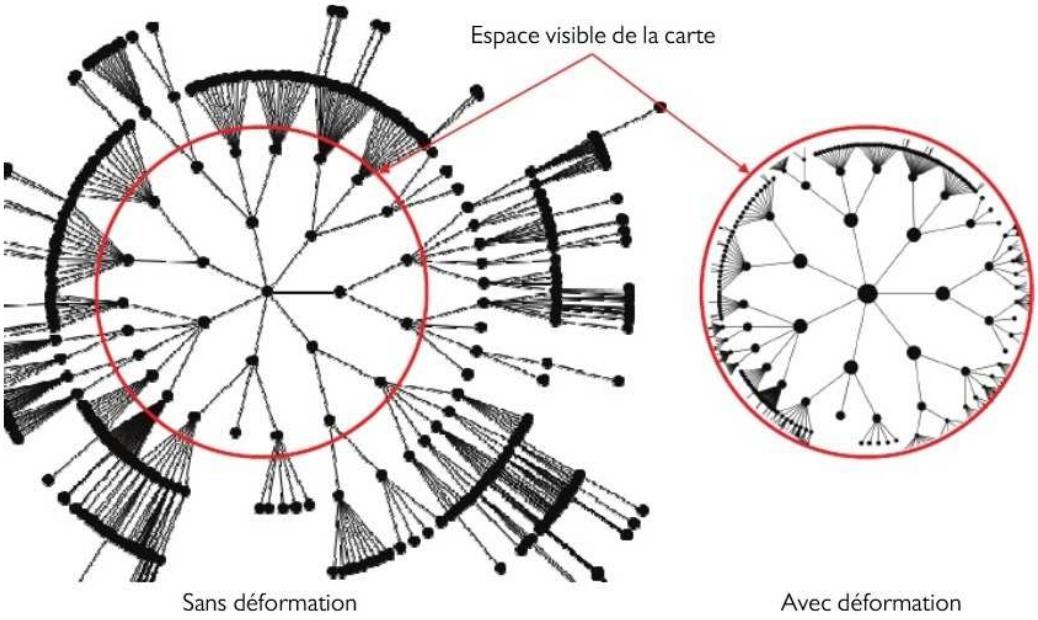


FIGURE A.21 – OS Eye Tree proposé par Tricot.

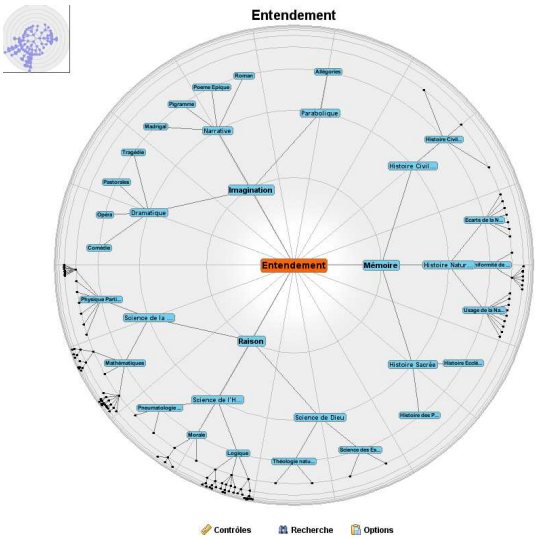


FIGURE A.22 – Visualisation d'une ontologie avec l'OS Eye Tree.

Tricot [Tricot, 2006] propose lui aussi un placement radial. Ce placement est complété par une vue déformante de type Fisheye polaire. Cette déformation illustrée en figure A.21 est basée sur une fonction de déformation de type tangente exponentielle qui « aplatit » l'infini. La visualisation d'une ontologie est présentée dans la figure A.22. Cependant ce travail ne cible pas les nœuds visuels.

Les applications Flokoon (`http://www.flokoon.com`) et Image-swirl [Jing *et al.*, 2010] proposent une autre façon de parcourir des arbres de données multimédia similaires (figure A.23). La navigation repose sur un développement progressif de l'arbre en faisant apparaître les média qui sont proches du nœud courant.

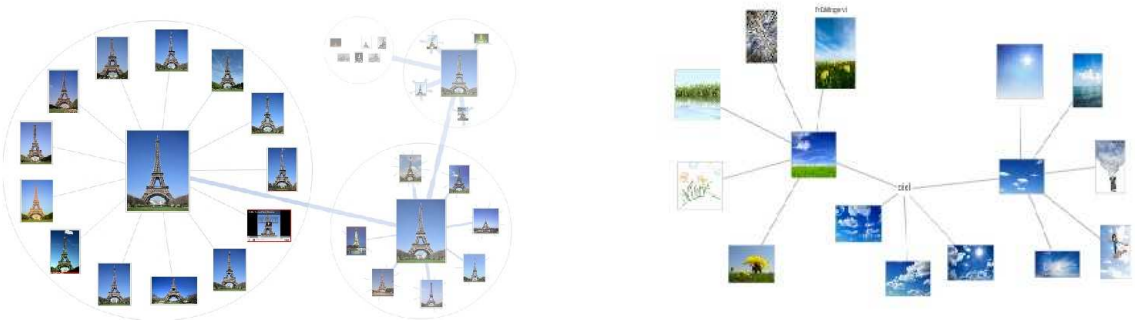


FIGURE A.23 – Navigation parmi des images similaires par parcours d'arbres. A gauche, Google Image Swirl de Google Inc. Mountain View, CA, USA. A droite, Flokoon (`http://www.flokoon.com`).

Une autre façon de visualiser et d'explorer des arbres est illustrée par le logiciel Photomesa qui est une implémentation des Tree-maps proposés par Bederson [Bederson *et al.*, 2002]. Les Tree-maps permettent de naviguer visuellement dans une hiérarchie de dossiers images tout en pouvant sélectionner l'aperçu de plusieurs répertoires dans la même fenêtre. Le fonctionnement repose sur un remplissage rectangulaire de la fenêtre du navigateur comme illustré dans la figure A.24.

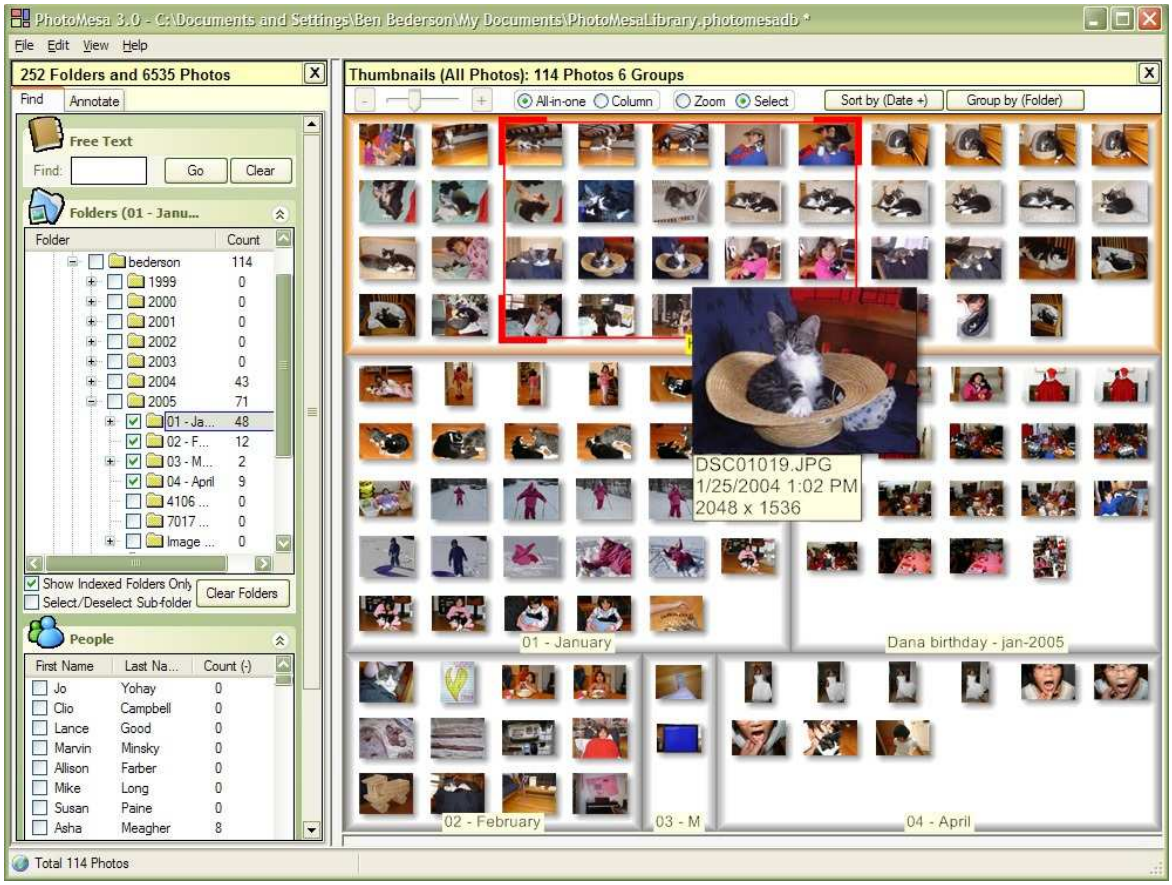


FIGURE A.24 – Le logiciel Photomesa (http://www.photomesa.com).

En synthèse, ces structures orientées liaisons nécessitent en général une structuration des données grâce à des méthodes de classification (hiérarchiques ou non). Ces méthodes peuvent parfois inclure une projection et une création de graphe des plus proches voisins.

A.3 Conclusion

Les visualisations existantes, dont les principales sont données dans cette annexe sont nombreuses. Nous pouvons en déduire que les représentations envisageables sont elles aussi nombreuses et que nous ne pouvons pas en donner une liste exhaustive ou une classification. Mais nous pouvons donner une liste non limitative :

- Grille 2D
- Grille sphérique
- Pellicule 2D
- Pellicule 3D
- Carrousel
- Placement 1D
- Placement 2D
- Placement 3D

- Placement sphérique
- Placement en graphe radial
- Placement en graphe développé
- Remplissage rectangulaire ...

Bibliographie

[Anderson *et al.*, 1990] Anderson, E., Bai, Z., Dongarra, J., Greenbaum, A., McKenney, A., Du Croz, J., Hammerling, S., Demmel, J., Bischof, C. et Sorensen, D. (1990). Lapack : A portable linear algebra library for high-performance computers. *In Proceedings of the 1990 ACM/IEEE Conference on Supercomputing*, Supercomputing '90, pages 2–11, Los Alamitos, CA, USA. IEEE Computer Society Press. *Citée à la page 102.*

[Azran, 2006] Azran, A. (2006). Spectral methods for automatic multiscale data clustering. *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR.* *Citée à la page 35.*

[Basu *et al.*, 2002] Basu, S., Banerjee, A. et Mooney, R. J. (2002). Semi-supervised clustering by seeding. *In Proceedings of the Nineteenth International Conference on Machine Learning*, ICML '02, pages 27–34, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc. *Citée à la page 40.*

[Bederson *et al.*, 2002] Bederson, B. B., Shneiderman, B. et Wattenberg, M. (2002). Ordered and quantum treemaps : Making effective use of 2d space to display hierarchies. *ACM Trans. Graph.*, 21(4):833–854. *Citée à la page 131.*

[Belkin et Niyogi, 2003] Belkin, M. et Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, 15(6):1373–1396. *Citée aux pages 28 and 50.*

[Bengio *et al.*, 2015] Bengio, Y., Goodfellow, I. J. et Courville, A. (2015). Deep learning. Book in preparation for MIT Press. *Citée à la page 28.*

[Benoit *et al.*, 2011] Benoit, A., Ciobotaru, M., Lambert, P. et Ionescu, B. (2011). Similarity measurement for animation movies. *In Proceedings of the 17th international conference on Advances in multimedia modeling - Volume Part I*, MMM'11, pages 350–358, Berlin, Heidelberg. Springer-Verlag. *Citée aux pages 52, 55, 57, and 62.*

[Bentley, 1975] Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517. *Citée à la page 37.*

[Bilenko *et al.*, 2004] Bilenko, M., Basu, S. et Mooney, R. J. (2004). Integrating constraints and metric learning in semi-supervised clustering. *In Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 11–, New York, NY, USA. ACM. *Citée aux pages 41 and 88.*

[Blum et Mitchell, 1998] Blum, A. et Mitchell, T. (1998). Combining labeled and unlabeled data with co–training. *In Proceedings of the Workshop on Computational Learning Theory.* *Citée à la page 40.*

[Borg et Groenen, 2005] Borg, I. et Groenen, P. (2005). *Modern Multidimensional Scaling : Theory and Applications.* Springer. *Citée à la page 25.*

[Boutin, 2005] Boutin, F. (2005). *Filtrage, classification et visualisation multi-échelles de graphes d'interactions à partir d'un focus.* Thèse de doctorat, Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier. *Citée aux pages xi, 128, and 129.*

[Breiman, 2001] Breiman, L. (2001). Random forests. *Mach. Learn.*, 45(1):5–32. *Citée aux pages 38 and 39.*

[Brin et Page, 1998] Brin, S. et Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107–117. *Citée à la page 120.*

[Brucker et Barthélemy, 2007] Brucker, F. et Barthélemy, J.-P. (2007). *Eléments de classification : Aspects combinatoires et algorithmiques.* Hermes Science Publication, Lavoisier. *Citée aux pages 24, 30, 31, 35, 36, and 37.*

[Bruley et Genoud, ntes] Bruley, C. et Genoud, P. (Dixièmes journées francophones sur l'Interaction Homme Machine, IHM 98, Nantes). Contribution à une taxonomie des représentations graphiques de l'information. *In 1998.* *Citée à la page 9.*

[Camargo et González, 2009] Camargo, J. et González, F. (2009). Visualization, summarization and exploration of large collections of images : State of the art. *In Latin-American Conference On Networked and Electronic Media. LACNEM.* *Citée aux pages 6, 19, and 29.*

[Card, 2007] Card, S. (2007). *Information Visualization*, chapitre 26. HCI Handbook. *Citée à la page 7.*

[Chen, 2010] Chen, C. (2010). Information visualization. *Wiley Interdisciplinary Reviews : Computational Statistics*, 2:387–403. *Citée à la page 8.*

[Chen *et al.*, 2000] Chen, J.-Y., Bouman, C. A. et Dalton, J. C. (2000). Hierarchical browsing and search of large image databases. *Trans. Img. Proc.*, 9(3):442–455. *Citée aux pages vii and 30.*

[Cockburn *et al.*, 2009] Cockburn, A., Karlson, A. et Bederson, B. B. (2009). A review of overview+detail, zooming, and focus+context interfaces. *ACM Comput. Surv.*, 41(1): 2 :1–2 :31. *Citée à la page 9.*

[Datta *et al.*, 2008] Datta, R., Joshi, D., Li, J. et Wang, J. Z. (2008). Image retrieval : Ideas, influences, and trends of the new age. *ACM Comput. Surv.*, 40(2):5 :1–5 :60. *Citée à la page 51.*

[Davidson et Ravi, 2005] Davidson, I. et Ravi, S. S. (2005). Agglomerative hierarchical clustering with constraints : Theoretical and empirical results. *In Lecture notes in computer science*, pages 59–70. Springer. *Citée à la page 41.*

[Davidson *et al.*, 2006] Davidson, I., Wagstaff, K. L. et Basu, S. (2006). Measuring constraint-set utility for partitional clustering algorithms. *In In : Proceedings of the Tenth European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 115–126. Springer. *Citée aux pages 42 and 44.*

[Demiriz *et al.*, 1999] Demiriz, A., Bennett, K. P. et Embrechts, M. J. (1999). Semi-supervised clustering using genetic algorithms. *In In Artificial Neural Networks in Engineering*, pages 809–814. ASME Press. *Citée à la page 41.*

[Donoho et Grimes, 2003] Donoho, D. L. et Grimes, C. (2003). Hessian eigenmaps : Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 100(10):5591–5596. *Citée à la page 29.*

[Eisermann, 2008] Eisermann, M. (2008). Comment fonctionne google ? *Quadrature*, 68:1–15. *Citée à la page 120.*

[Gomi *et al.*, 2008] Gomi, A., Miyazaki, R., Itoh, T. et Li, J. (2008). Cat : A hierarchical image browser using a rectangle packing technique. *2010 14th International Conference Information Visualisation*, 0:82–87. *Citée aux pages vii and 13.*

[Grabisch *et al.*, 2008] Grabisch, M., Kojadinovic, I. et Meyer, P. (2008). A review of capacity identification methods for Choquet integral based multi-attribute utility theory,

Applications of the Kappalab R package. *European journal of operational research*, 186(2): 766 – 785. *Citée à la page 57.*

[Hastie *et al.*, 2009] Hastie, T. J., Tibshirani, R. J. et Friedman, J. H. (2009). *The elements of statistical learning : data mining, inference, and prediction*. Springer series in statistics. Springer, New York. Autres impressions : 2011 (corr.), 2013 (7e corr.). *Citée aux pages 37 and 38.*

[Hoffer et Ailon, 2014] Hoffer, E. et Ailon, N. (2014). Deep metric learning using triplet network. *CoRR*, abs/1412.6622. *Citée à la page 29.*

[Hubert et Arabie, 1985] Hubert, L. et Arabie, P. (1985). Comparing partitions. *Journal of classification*, 2(1):193–218. *Citée aux pages 33 and 60.*

[Ionescu *et al.*, 2012] Ionescu, B., Mironica, I., Seyerlehner, K., Knees, P., Schlüter, J., Schedl, M., Cucu, H., Buzo, A. et Lambert, P. (2012). Arf @ mediaeval 2012 : Multimodal video classification. *In* Larson, M. A., Schmiedeke, S., Kelm, P., Rae, A., Mezaris, V., Piatrik, T., Soleymani, M., Metze, F. et Jones, G. J. F., éditeurs : *MediaEval*, volume 927 de *CEUR Workshop Proceedings*. CEUR-WS.org. *Citée aux pages 52 and 86.*

[Ionescu, 2007] Ionescu, B.-E. (2007). *Caractérisation symbolique de séquences d'images : application aux films d'animation*. Thèse de doctorat, Laboratoire d'Informatique, Systèmes, Traitement de l'Information et de la Connaissance (LISTIC) en cotutelle avec l'Universitatea politehnica Bucuresti. *Citée à la page 57.*

[Jaeschke *et al.*, 2005] Jaeschke, G., Leissler, M. et Hemmje, M. (2005). Modeling interactive, 3-dimensional information visualizations supporting information seeking behaviors. *In Lecture Notes in Computer Science*. *Citée à la page 9.*

[Jain, 2010] Jain, A. K. (2010). Data clustering : 50 years beyond k-means. *Pattern Recogn. Lett.*, 31(8):651–666. *Citée aux pages viii and 39.*

[Jankun-Kelly *et al.*, 2006] Jankun-Kelly, T., Kosara, R., Kindlmann, G., North, C., Ware, C. et Bethel, E. W. (2006). Is there science in visualization. *In IEEE Visualization Conference Compendium*. *Citée à la page 5.*

[Jankun-Kelly et Ma, 2003] Jankun-Kelly, T. J. et Ma, K.-L. (2003). Moiregraphs : radial focus+context visualization and interaction for graphs with visual nodes. *In Proceedings of the Ninth annual IEEE conference on Information visualization*, INFOVIS'03, pages 59–66, Washington, DC, USA. IEEE Computer Society. *Citée aux pages xi, 128, and 129.*

[Jing *et al.*, 2010] Jing, Y., Rowley, H. A., Rosenberg, C., Wang, J., Zhao, M. et Covell, M. (2010). Google image swirl, a large-scale content-based image browsing system. *2012 IEEE International Conference on Multimedia and Expo*, 0:267. *Citée à la page 131.*

[Joachims, 1999] Joachims, T. (1999). Transductive inference for text classification using support vector machines. *In Proceedings of the Sixteenth International Conference on Machine Learning*, ICML '99, pages 200–209, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc. *Citée à la page 40.*

[Kamvar *et al.*, 2003] Kamvar, S. D., Klein, D. et Manning, C. D. (2003). Spectral learning. *In In IJCAI*, pages 561–566. *Citée à la page 87.*

[Kendall et Gibbons, 1990] Kendall et Gibbons (1990). *Rank Correlation methods*. Edward Arnold, London. *Citée à la page 59.*

[Kohonen *et al.*, 2001] Kohonen, T., Schroeder, M. R. et Huang, T. S., éditeurs (2001). *Self-Organizing Maps*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 3rd édition. *Citée à la page 27.*

[Law *et al.*, 2013] Law, M. T., Thome, N. et Cord, M. (2013). Quadruplet-Wise Image Similarity Learning. *In IEEE International Conference on Computer Vision (ICCV)*, pages 249 – 256, Sydney, Australia. *Citée à la page 39.*

[Lee et Verleysen, 2002] Lee, J. A. et Verleysen, M. (2002). Nonlinear projection with the isotop method. *In Proceedings of the International Conference on Artificial Neural Networks*, ICANN '02, pages 933–938, London, UK, UK. Springer-Verlag. *Citée aux pages vii, 27, and 28.*

[Lespinats *et al.*, 2007] Lespinats, S., Verleysen, M., Giron, A. et Fertil, G. (2007). Dd-hds : A method for visualization and exploration of high-dimensional data. *Trans. Neur. Netw.*, 18(5):1265–1279. *Citée à la page 29.*

[Li *et al.*, 2009] Li, Z., Liu, J. et Tang, X. (2009). Constrained clustering via spectral regularization. *In 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 421–428. *Citée à la page 87.*

[Liu *et al.*, 2004] Liu, H., Xie, X., Tang, X., Li, Z.-W. et Ma, W.-Y. (2004). Effective browsing of web image search results. *In Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval*, MIR '04, pages 84–90, New York, NY, USA. ACM. *Citée aux pages vii, 10, 11, 12, 14, and 125.*

[Ludwig *et al.*, 2009] Ludwig, O., Delgado, D., Goncalves, V. et Nunes, U. (2009). Trainable classifier-fusion schemes : An application to pedestrian detection. *In IEEE Int. Conf. On Intelligent Transportation Systems*, pages 432–437. *Citée à la page 78.*

[Maimon et Rokach, 2005] Maimon, O. et Rokach, L. (2005). *Data Mining and Knowledge Discovery Handbook*. Springer-Verlag New York, Inc., Secaucus, NJ, USA. *Citée à la page 36.*

[Mallapragada *et al.*, 2008] Mallapragada, P. K., Jin, R. et Jain, A. K. (2008). Active query selection for semi-supervised clustering. *In 19th International Conference on Pattern Recognition (ICPR 2008), December 8-11, 2008, Tampa, Florida, USA*, pages 1–4. *Citée à la page 94.*

[Maximo *et al.*, 2009] Maximo, A., Saba, M. P. et Velho, L. (2009). M-cube : A visualization tool for multi-dimensional multimedia databases. *In Proceedings of Interaction.* *Citée à la page 125.*

[Mironica *et al.*, 2013] Mironica, Ionut andIonescu, B., Knees, P. et Lambert, P. (2013). An in-depth evaluation of multimodal video genre categorization. *In MediaEval.* *Citée aux pages 77, 81, and 106.*

[Mohammadi-Aragh et Jankun-Kelly, 2005] Mohammadi-Aragh, M. J. et Jankun-Kelly, T. J. (2005). Moiretrees : visualization and interaction for multi-hierarchical data. *In Proceedings of the Seventh Joint Eurographics / IEEE VGTC conference on Visualization*, EUROVIS'05, pages 231–238, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association. *Citée à la page 130.*

[Mohar, 1997] Mohar, B. (1997). Some applications of laplace eigenvalues of graphs. *In GRAPH SYMMETRY : ALGEBRAIC METHODS AND APPLICATIONS, VOLUME 497 OF NATO ASI SERIES C*, pages 227–275. Kluwer. *Citée aux pages 45 and 47.*

[Pearson, 1901] Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2:559–572. *Citée aux pages vii and 25.*

[Pfitzner *et al.*, 2003] Pfitzner, D., Hobbs, V. et Powers, D. (2003). A unified taxonomic framework for information visualization. *In APVis '03 Proceedings of the Asia-Pacific symposium on Information visualisation - Volume 24 - Pages 57-66.* *Citée à la page 6.*

[Podani, 1997] Podani, J. (1997). A measure of discordance for partially ranked data when presence/absence is also meaningful. *Coenoses*, 12:127–130. *Citée à la page 62.*

[Rangapuram et Hein, 2012] Rangapuram, S. S. et Hein, M. (2012). Constrained 1-spectral clustering. *In Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2012, La Palma, Canary Islands, April 21-23, 2012*, pages 1143–1151. *Citée aux pages x, 87, 94, 97, 102, and 111.*

[Rifqi *et al.*, 2008] Rifqi, M., Lesot, M.-J. et Detyniecki, M. (2008). Fuzzy order-equivalence for similarity measures. *In* CNF, I., éditeur : *NAFIPS*. *Citée à la page 59.*

[Roweis et Saul, 2000] Roweis, S. T. et Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE*, 290:2323–2326. *Citée à la page 28.*

[Schmiedeke *et al.*, 2012] Schmiedeke, S., Kofler, C. et Ferrané, I. (2012). Overview of the mediaeval 2012 tagging task. *In* Larson, M. A., Schmiedeke, S., Kelm, P., Rae, A., Mezaris, V., Piatrik, T., Soleymani, M., Metze, F. et Jones, G. J. F., éditeurs : *MediaEval*, volume 927 de *CEUR Workshop Proceedings*. CEUR-WS.org. *Citée aux pages 53, 71, 76, and 78.*

[Schmiedeke *et al.*, 2013] Schmiedeke, S., Xu, P., Ferrané, I., Eskevich, M., Kofler, C., Larson, M., Estève, Y., Lamel, L., Jones, G. et Sikora, T. (2013). Blip10000 : A social video dataset containing spug content for tagging and retrieval. *ACM Multimedia Systems Conference*. *Citée aux pages 102 and 106.*

[Shi et Malik, 2000] Shi, J. et Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905. *Citée à la page 49.*

[Shneiderman, 1996] Shneiderman, B. (1996). The eyes have it : a task by data type taxonomy for information visualizations. *In IEEE Visual Languages*. *Citée aux pages vii, 8, and 9.*

[Smeaton *et al.*, 2006] Smeaton, A. F., Over, P. et Kraaij, W. (2006). Evaluation campaigns and trecvid. *In MIR '06 : Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, pages 321–330, New York, NY, USA. ACM Press. *Citée à la page 52.*

[Tenenbaum *et al.*, 2000] Tenenbaum, J. B., de Silva, V. et Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500): 2319–2323. *Citée aux pages vii, 26, and 27.*

[Torgerson, 1952] Torgerson, W. S. (1952). Multidimensional scaling : I. theory and method. *Psychometrika*, 17:401–419. *Citée à la page 26.*

[Tory et Möller, 2002] Tory, M. et Möller, T. (2002). A model-based visualization taxonomy. *In School of Computing Science, Simon Fraser University*. *Citée à la page 9.*

[Tricot, 2006] Tricot, C. (2006). *Cartographie sémantique*. Thèse de doctorat, Université de Savoie. *Citée aux pages vii, 7, 8, 10, and 131.*

[van der Maaten et Hinton, 2008] van der Maaten, L. et Hinton, G. (2008). Visualizing high-dimensional data using t-sne. *Citée à la page 29.*

[Villerd, 2008] Villerd, J. (2008). *Représentations visuelles adaptatives de connaissances associant projection multidimensionnelle (MDS) et analyse de concepts formels (FCA)*. Thèse de doctorat, École doctorale ICMS de l'École des Mines de Paris. *Citée à la page 5.*

[Voiron *et al.*, 2012] Voiron, N., Benoit, A. et Lambert, P. (2012). Automatic difference measure between movies using dissimilarity measure fusion and rank correlation coefficients. *In Content-Based Multimedia Indexing (CBMI), 10th International Workshop*. *Citée à la page 52.*

[von Luxburg, 2007] von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416. *Citée aux pages 43, 44, 46, 48, 49, and 50.*

[Vu *et al.*, 2012] Vu, V., Labroche, N. et Bouchon-Meunier, B. (2012). Improving constrained clustering with active query selection. *Pattern Recognition*, 45(4):1749–1758. *Citée aux pages 42, 86, and 114.*

[Wagstaff *et al.*, 2001] Wagstaff, K., Cardie, C., Rogers, S. et Schroedl, S. (2001). Constrained k-means clustering with background knowledge. *In In ICML*, pages 577–584. Morgan Kaufmann. *Citée aux pages 41 and 88.*

[Wang et Davidson, 2010] Wang, X. et Davidson, I. (2010). Flexible constrained spectral clustering. *In Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*, pages 563–572. *Citée à la page 87.*

[Xiong *et al.*, 2014] Xiong, C., Johnson, D. M. et Corso, J. J. (2014). Active clustering with model-based uncertainty reduction. *CoRR*, abs/1402.1783. *Citée aux pages 43, 50, 86, 87, 102, 103, and 114.*

[Xu *et al.*, 2009] Xu, L., Li, W. et Schuurmans, D. (2009). Fast normalized cut with linear constraints. *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 0:2866–2873. *Citée à la page 87.*